Matching Literature Heritage Entities From Heterogeneous Data Sources Based On The Textual Description

Georgii Sipovskii
ITMO University
Saint-Petersburg, Russia
gsipovskii@gmail.com

Nikolay Teslya
SPC RAS
Saint-Petersburg, Russia
teslya@iias.spb.su

***Abstract*—The paper focuses on problem of short text matching for literature heritage entities alignment from heterogeneous data sources. The overview of existing methods showed that all of them works well for long texts. The paper proposes modification of Jacquard similarity metric for solving the problem based on similarity of unique text tokens adjusted to the specifics of literature heritage domain. Achieved results were evaluated on the literature heritage of the A.S. Pushkin gathered from the various heterogeneous sources (datasets, full works compilations. Encyclopedia of A.S. Pushkin) and shown high accuracy of finding corresponding entities within the system by developed method.**

I. Introduction

Work with relatively small texts is still an important tasks in many information systems. These texts could be titles of entities in the systems, descriptions, characteristics names, comments, etc. Analyzing such texts allows to identify possible duplicates in datasets and perform matches, compare entities describing the same real-world objects [1], optimize the cache size of search queries.

In this work we will concentrate on the matching objects in database based on short text descriptions. The source of the task lies in the field of processing literary heritage of the great Russian poet A.S. Pushkin. There are a lot of heterogeneous information sources related both to the works by A.S. Pushkin and to himself. Examples of these sources are Pushkin's complete collection of works (i.e. Pushkin's complete collection of works in 16 volumes [2]) that also contains variants and editions of texts and scientific comments related to each work, Encyclopedia of Pushkin's works [3]. Beside these sources there are several datasets related to the Pushkin works, such as Index of works [4] and Full text of poems [5]. All data sources are created in very different times and the same entities could be titled or described in different way. In example the same poem could be known under several title, as well as poem's text itself could differ based on the source of the poem (first of last edition of the poem, of research result on the handwriting or historical information around it).

This research is aimed to provide accurate matching between all entities from the sources mentioned above taking into account differences between data presented in sources. The matching is based on comparison between the short text such as titles and first lines of entity's text (incipit) from all data sources as far as only these data are shared between all the data sources. As the result we developed a method that compares entities based on their text characteristics focusing on unique elements that these entities share.

Particular interest of the problem of short text matching lies in the aspect of structure of the processed text data itself which can not be considered as standard, hence require specific approach for matching entities these texts represent. Those intricacies will be reviewed in the according section of this article.

The rest of the paper is structured as follows. Section II provides overview on existing text matching approaches. Section III overviews the problem statement and describes data sources. Section IV describes the method proposed to solve the problem. The evaluation of the proposed method and discussion on results are presented in Section V. The last Section VIprovides conclusion and directions of future work.

II. Related work

The problem of short text matching has been studied for approximately 40 years. There are a lot of methodologies and approaches developed [6] to solve it. The diagram in the Fig. 1. illustrates major types of methods in a structured manner.

A. String comparison methods

String methods of measuring the similarity of texts perceive text only as a sequence of characters, ignoring their lexical features. The list below contains the most known methods belonging to this category:

- *LCS (Largest Common Substring)* – uses the calculation of the maximum occurrence of a substring in a string to determine the similarity of two strings
- *N – grams* – considers a string as a combination of all possible substrings of length N and compares the number of entries of one string relative to another.
- *Jaccard similarity* [7] – calculates the distance between strings by the ratio of their matched characters and

the sum of characters of these strings from which the numerator is subtracted.

— *Cosine similarity* is a measure of similarity between two vectors of the inner product space, which measures the cosine of the angle between. This method of measuring similarity is rarely used as-is, but as a numerical measure of similarity between vectors in Corpus and Neural network-based methods.

Ideas behind other methods of string comparison such as Levenstein Distance [8], Affine gap distance and others will not be discussed in this section, but most of them are covered in [9], the same applies for methods from other groups.

B. Corpus methods

Corpus is a processed extensive and structured set of texts. Corpus methods use various statistical methods and approaches for processing this corpus of texts to compare them, most popular of those are presented below:

— *LSA (Latent Semantic Analysis)* [10] – analyzes the corpus of texts and creates a matrix of words and documents. In practice, singular value decomposition (SVD) is used to reduce document space while maintaining similarity between lines. The similarity between the strings is measured by calculating the scalar product or the cosine distance between the vectors.

— *LDA (Latent Dirichlet Allocation)* [11] is a three–level Bayesian probability model. It models the topic of the text based on the content of the text, and then compares the texts by their respective topic distributions.

— *HAL (Hyperspace Analogue to Language)* [12] – uses lexical coincidences to create a multidimensional semantic space. As a part of corpus processing, a word-by-word matrix is created in which columns with low entropy are omitted. This calculates it according to the distance of the words to the selected keyword. HAL also considers word order information by assigning matrix values based on the position of the word relative to the keyword.

— *PMI-IR (Pointwise Mutual Information - Information Retrieval)* [13] – this approach utilizes Advanced Search query syntax to calculate probability of words similarity. The idea of score calculation is simple: the more often compared words appear together on the same web page, the higher their PMI-IR score. Google has developed its own interpretation of this approach called NGD [14]. This technique was proved to be effective when combined with already existed methods of this category was able to improve existing approaches i.e. GLSA [15] and SCO-PMI [16] combined it with latent semantic analysis to foster performance of the original method.

C. Neural network methods

Neural network word processing models use representations of texts in the form of word vectors formed from processed texts that make up the vector space of word features. This vector space is called word embeddings. This vector space can be constructed using several standard models: Word2Vec, CBOW, and Skip-gram. The very same comparison of texts takes place by means of using these embeddings by various text processing models, and the enumeration below will be devoted to them

— *ETM (Embedding-based Topic Model)* [17] – This model uses embeddings to find key topics for a collection of short texts. It uses embeddings to combine short texts into large artificial texts, inside which the K-mean is measured to calculate the distance between the texts.

— *DSSM (Deep Structured Semantic Models)* [18] is a family of neural networks that build vector feature spaces from texts and use cosine distance to measure the distance between texts. There are 3 main types of models in this family for building a vector space:

  — *Feed-Forward Network-Based model* – in this type of DSSM, the text is represented in bag-of-3-grams, which are then converted into Boolean vectors and are fed to the feed-forward network input, the output is a vector that is a representation of the vector relative to which the closest texts will be identified.

  — *Convolutional model* – this modification of DSSM uses a sliding window to build n-grams of words, i.e. each word is represented by a Boolean vector of occurrence of n-gram. These n-grams are combined into a feature vector, to which the convolution oper-

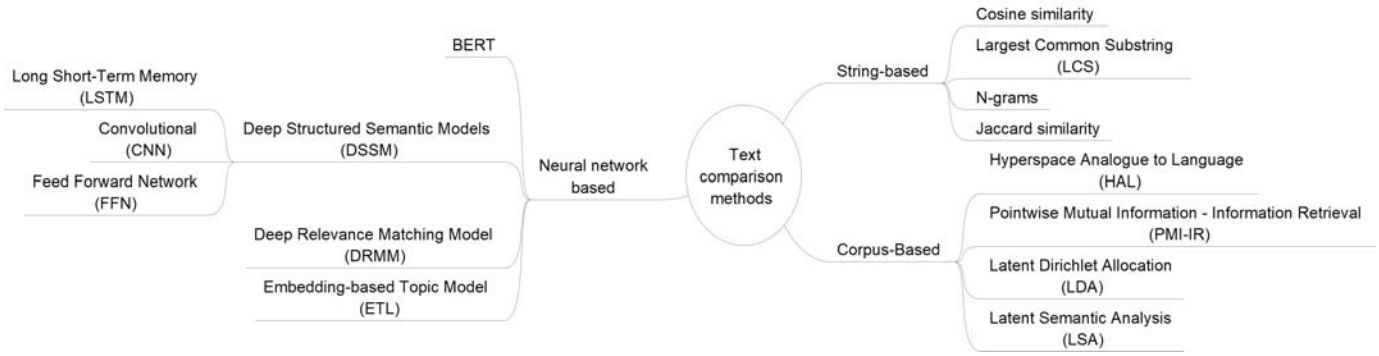

Fig. 1. Types of text comparison methods

ation is then applied. Result of this transformation is fed to the feed-forward network input, then a max-pooling layer takes the feature vector from the last layer (it takes the maxima of the vectors of all sliding text windows) after which max-pooling operation is applied again to the previous result and the feature vector of the input text.

— *LSTM (Long Short-Term Memory)* – this modification of DSSM uses LSTM to construct a feature vector, unlike convolution, this is the only significant architectural difference, apart from using cosine distance to measure the proximity of texts. This modification was significantly revised in [19], where a new normalizing algorithm was used. It helped to cope with the problem of vanishing gradient on long text sequences, characteristic of this type of models.

— *DRMM (Deep Relevance Matching Model)* – in this type of model, a matrix of word interaction is created from which a high-level feature is then extracted. To get the weight of each word in a sentence, a term gate is applied to it, the sum of these weights is one. Next, for each pair of words, the cosine distance is calculated, which lies in the range from -1 to 1, which is divided into several intervals. Afterwards, for each pair the cosine distance values are calculated. The number of words in each interval is the value of the feature vector, which represents the assessment of the correspondence of each word to the second sentence. The final text compliance score is the weighted sum of the compliance scores of each word-per-sentence.

— *BERT* [20] – this model is based on the transformer architecture [21], unlike previous models, it does not use recurrent neural networks. BERT is a trained text encoder (this encoder trains on a text corpus without supervision), which encodes the input text into a feature vector. The encoder's training most often consists in trying to predict randomly masked words at the input based on the output feature vector, or in predicting whether sentences are conjugate in the original corpus. This type of model has shown better efficiency in the task of comparing texts, thanks to the attention mechanism, than its predecessors, it is not surprising that this architecture has become the basis for current state-of-the-art solutions in the field of natural language processing.

III. Problem statement

Within the scope of the research, entities representing the literature heritage by the great Russian poet A.S. Pushkin are considered. The heritage unites not only the texts of works but also all references, mentions, discussions, historical research, critics and other materials related to the works and the author. Examples of heritage entities for the A.S. Pushkin are complete collection of works that also contains variants and editions of texts, scientific comments related to each work, Encyclopedia of Pushkin's works [3]. There are also datasets created by researchers of A.S. Pushkin's life and works, e.g. Index of works [4] or Full text of poems [5]. Even though the common nature of all entities in datasets is obvious to a person, an information system needs to describe an method for matching these entities for automatic and accurate alignment of them. This study was performed on sets of comparable entities with more than a thousand of exemplars each.

A. Overview of compared entities

In this research we had used entities, representing works by A.S. Pushkin that were obtained from the several sources, like FEB-web portal (Text), Index of works and letters of Pushkin A. S. [4] (Index), Encyclopedia of A.S. Pushkin. The problem with entities matching is that in different sources the same entities could be titles in different way. The dataset of indexes of A.S. Pushkin's works was taken as the main source of the works titles [4].

B. Comparison characteristics

As mentioned earlier, two entities are compared: the Index and the text of the writer. Each Index item is unique and represents an artifact of the poet's creativity and life activity behind his authorship such as: a poem, a novel, an essay and other less frequent types of works. Entity of the Index has two significant fields within the scope of this task under consideration: the title and the incipit (a few words from the text used to uniquely identify it), the other fields of this entity do not carry textual information about it and only contain meta information.

The Text entity has only its name and qualitatively this field can contain an incipit and the title of the work. There is no specific order of inclusion, and there are no rules of it either. It is also worth noting that one work may correspond to one or more texts (in one case, 7 variations of one text were identified) entities. A distinctive feature that both of these characteristics share is that the most of names featured in them are common names, proper names, addresses, and descriptions. All of them are not complete sentences, moreover, the Index and Text items are not directly related to each other and are considered as separate works of the poet.

Compared entities mostly contain texts in Russian language, although due to certain pieces being written in French can be found, their names and incipits are in French respectfully. For completeness of compared entities' qualitative overview a couple of examples of the works and corresponding texts in French and Russian languages is provided in the Table I. Selected examples represent previously discussed specifics and nuances of compared entities encountered within the scope of the problem.

C. Characteristics of text data

For better understanding of given data from the point of view of statistics it is essential to construct an overview of it with visualization. It will allow to acknowledge numerically which pieces of text data will be compared.

The following data distributions give an insight on compared entities:

TABLE I. EXAMPLES OF WORKS IN RUSSIAN AND FRENCH

Chracteristics	Entity in Russian	Entity in French
Work title	"К. А. Тимашевой" "To K. A. Timasheva"	"Je chante ce combat que Toly remporta..." "I sing of the battle Toly won..."
Work incipit	"Я видел вас, я их читал" "I saw you, I read them"	
Text title	"Я видел вас, я их читал" (К. А. Тимашевой) "I saw you, I read them" (To K. A. Timasheva)	"Je chante ce combat que Toly remporta..." "I sing of the battle Toly won..."

— distribution of the number of tokens (Fig. 2)
— distribution of weighted symbolic length of tokens (length weight is the relative length of all tokens in the set of entity tokens) (Fig. 3)
— distribution of average square deviation of the number of characters in the token (Fig. 4).

Analyzing the distributions in the figures, the following conclusions were made:

— On average, the number of tokens in the texts for comparison is higher by one or two tokens than in the works.
— The graphs of weighted token lengths and standard deviation of entity tokens are similar to each other.

To summarize, the high-level view of comparison can be described as following: sets of tokens (from two to seven) of approximate lengths (from four to ten characters) with deviations (from one to three) in their characters are going to be compared most often. These texts are quite small, from fifteen to seventy characters per entity, which is a relatively small amount of textual information per sample of comparison.

Examined characteristics of processed texts, combined with the fact that given texts share no common context and semantics between each other and do not contain any high level language features, suggest that usage of corpus and neural network based approaches is not applicable in this case, and the basic approach of string comparison for comparing entities should be used.

IV. OVERVIEW OF PROPOSED METHOD

A. General approach

The general line of the proposed method is to use the Bag of Words model [22]. This model is usually used as input in corpus and neural approaches to construct a vector space of words and compare the unique elements of one set with another. For comparison only tokens of a length greater than one are used. Most often tokens of length one are auxiliary parts of speech (preposition, conjunction or pronoun) in Russian. For the domain considered in this work such token do not carry a large amount of information both from the point of view of describing the object or subject of the described work, and from the point of view of symbolic information. Even though they are used quite often in names or incipits. Another feature of Bag of Words is that it does not take into account the order of words in the source. In other circumstances, this could have been a disadvantage. But since in text entity the incipit and the name can both be represented in an indefinite order, therefore in this particular case it is justified. If compared texts can be described as semantically meaningful then the order of words would matter and method proposed in [23] can be used. In the process of comparing arrays of entities, entities are compared in pairs, the comparison method decides whether a given text corresponds to a given work or not.

B. Entity comparison method

Let us consider the process of comparing entities in more detail. It includes the following steps:

1) for the text data of each entity, an array of unique tokens is extracted.
2) the number of occurrences of each token for the entity is calculated
3) an entity with more unique tokens is identified, within the scope of comparison it will be considered superset.
4) the other entity in pair is considered to be a subset and after being compared with the superset it is defined if those sets share any tokens.
5) if there are none, then the entities are not compared further, otherwise the similarity coefficient of the entities is calculated.

The main difference between this approach and direct comparison is in the comparison of unique elements (tokens), not all of them. This difference allows us to compare entities only by the unique elements characterizing this entity. Let's demonstrate it by example from Table II:

TABLE II. EXAMPLE OF ENTITIES STRING CHARACTERISTICS

	Title	Incipit
Entity from the Work	"Соловей" *Solovey* (Nightingale)	
Entity from the Text	"Песни западных славян — Соловей", *Pesni zapadnykh slavyan - Solovey* (Songs of the Western Slavs - Nightingale)	"Соловей мой, соловейко..." *Solovey, moi, soloveiko...* (Nightingale, my nightingale...)

These two entities are related to the same poem by the poet and must be correlated by an the method. However, classical methods applied "as is" will show a small degree of similarity, although the fact of correspondence is obvious to a person. If we compare only the unique tokens of these entities, then the degree of correspondence will be 70%, which is already significant. It is worth to note that from a lexical point of view,

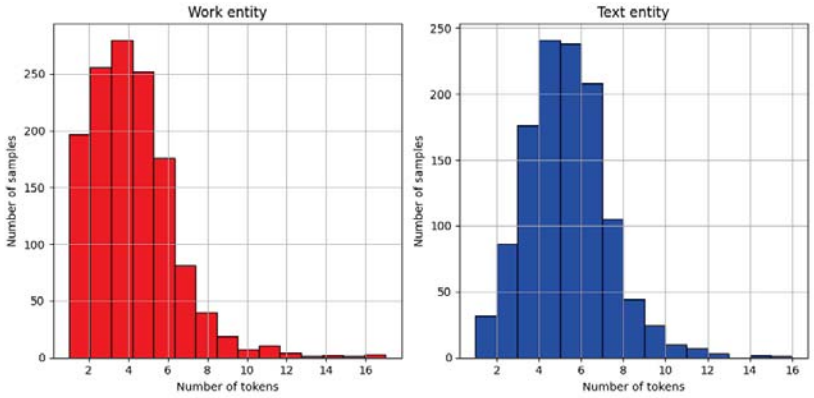

Fig. 2. Distribution of samples by number of tokens

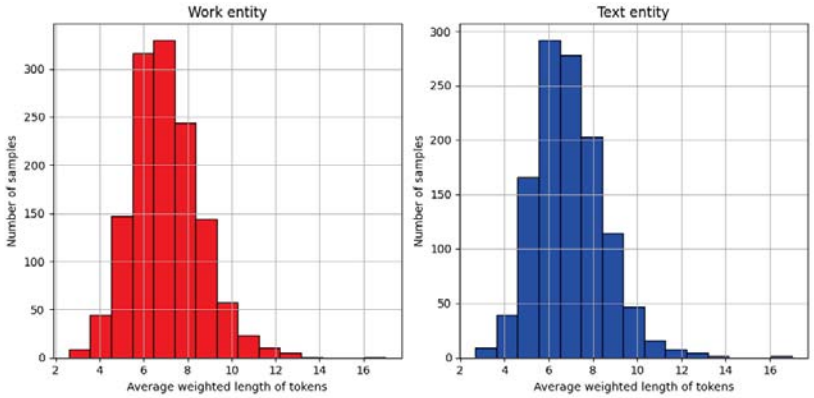

Fig. 3. Distribution of samples by weighted length of tokens

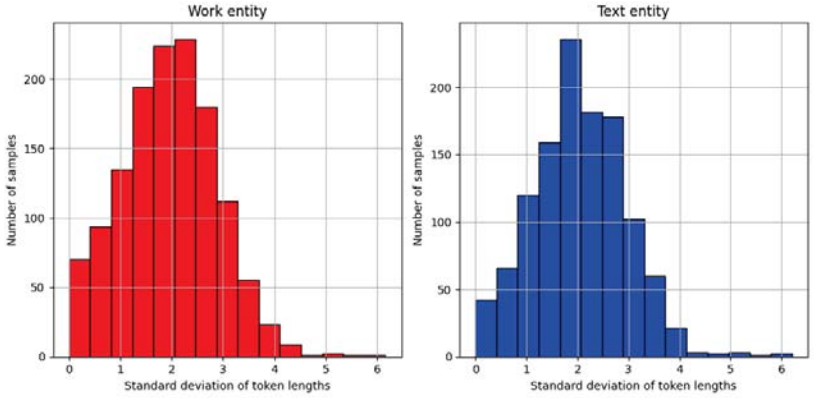

Fig. 4. Distribution of samples by standard deviation of token lengths

the pronoun ”my” only indicates a particular nightingale, but does not describe it in any way, therefore it does not help to distinguish it from other “nightingales”. This is part of the specifics of this task, since the titles of works often contain repetitive elements, and the names themselves are not related from a semantic point of view.

C. Similarity coefficient evaluation

The similarity coefficient of two sets of tokens is a modification of the Jaccard similarity, which is calculated not relative to the matched characters in the sequence, but relatively to the length of matched unique tokens in the sequence. This modification is demonstrated by the formulas below:

The standard Jaccard Similarity Formula:

$$J_c = \frac{A \wedge B}{A \vee B} = \frac{c}{a + b - c} \quad (1)$$

where [7]:
— a is the number of characters in the first string.
— b is the number of characters in the second string.
— c is the common matched characters in both strings.
Proposed modification:

$$J_{c_m} = \frac{len(A \cap B)}{len(A \cup B)} \quad (2)$$

where A and B are sets of unique entity tokens and A is a subset of B.

Tokens are considered matched if the following conditions are met:
— all characters of one token match all characters of another token.
— the tokens matched each other with the number of typos.

Only certain pairs of characters are considered as a typo, for example: ”k” and ”c” may be a typo, but ”g” and ”s” can not. These pairs of characters will be specific to each language, in this particular case they are presented due to possible human errors and different spelling rules of words in different time periods.

V. Evaluation and discussion

Schematically the described method is represented in the Fig. 6, only notable modification to it being the variable work_coefficient_max that ensures the best matches between sets of entities. Once this match is found, the same method is ran against other entities of the same kind to find duplicates within scope of one entity with a greater threshold value of 0.75.

Dataset for the comparison contained around 900 items in Index and 1300 items in Texts. If a correspondence between entities that were not presented in one of the sets was established they were counted as false negatives. Test was ran on dataset five times. The results are shown in the Table III. Additionally, entities were matched using Levenshtein (L_r) [24] and naive Jaccard similarity (J_c) [7] for comparison.

TABLE III. Entity matching result metrics

Metric	Value L_r	Value J_c	Value J_{c_m}
Specifity (true negative rate)	0.000	0.806	0.974
Sensitivity (true positive rate)	0.652	0.908	0.995
Accuracy	0.481	0.874	0.988
False positive rate	1.000	0.194	0.026
False negative rate	0.348	0.092	0.005

Such results can be explained by specifics of the task and approach to comparing text characteristics by the developed method. In the following Fig. 5, the dependence of the similarity coefficient relative to the number of matched tokens is presented. There is an explicit linear dependence on it, which is not surprising, given the formula for calculating the coefficient. Similarity values of less than 0.2 are cut off. This method turned out to be more efficient and more accurate than the naive Jaccard implementation, N-gramm and Levenshtein [24] distance, since it uses the specifics of the task domain and incorporates the strengths of this method group for accomplishing the task of identifying corresponding entities in the given dataset.

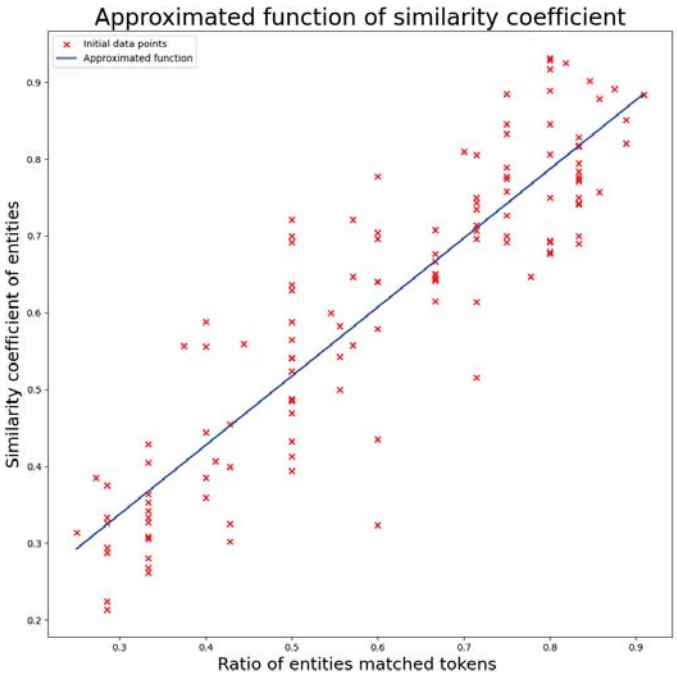

Fig. 5. Similarity coefficient function approximation

VI. Conclusion

The developed method is the first iteration of applying this approach in practice within the scope of the entities matching for literary heritage of A.S. Pushkin and is used for similar tasks in the system. The method provides good results in terms of specifity and sensivity as well as accuracy is quite high.

It is necessary to mention that currently proposed method without any modifications will try to compare entities of different languages which is inefficient, since a work of any author has a language of origin before translation to other languages. Topic of semantic errors during translation process is certainly out of the scope of this paper, so it is going to be noted that language identification of compared entities before comparison operation may improve efficiency of this method. Another limitation of this method comes from the nature of the writing systems used in different languages. This method will perform poorly for works of languages that use Logographic writing systems for example, because the number of symbols representing a real-world entity in the such languages is considerably lower. On the contrary, the amount of those symbols themselves varies much greater than in languages that use alphabets. For instance, handling of typos in case of

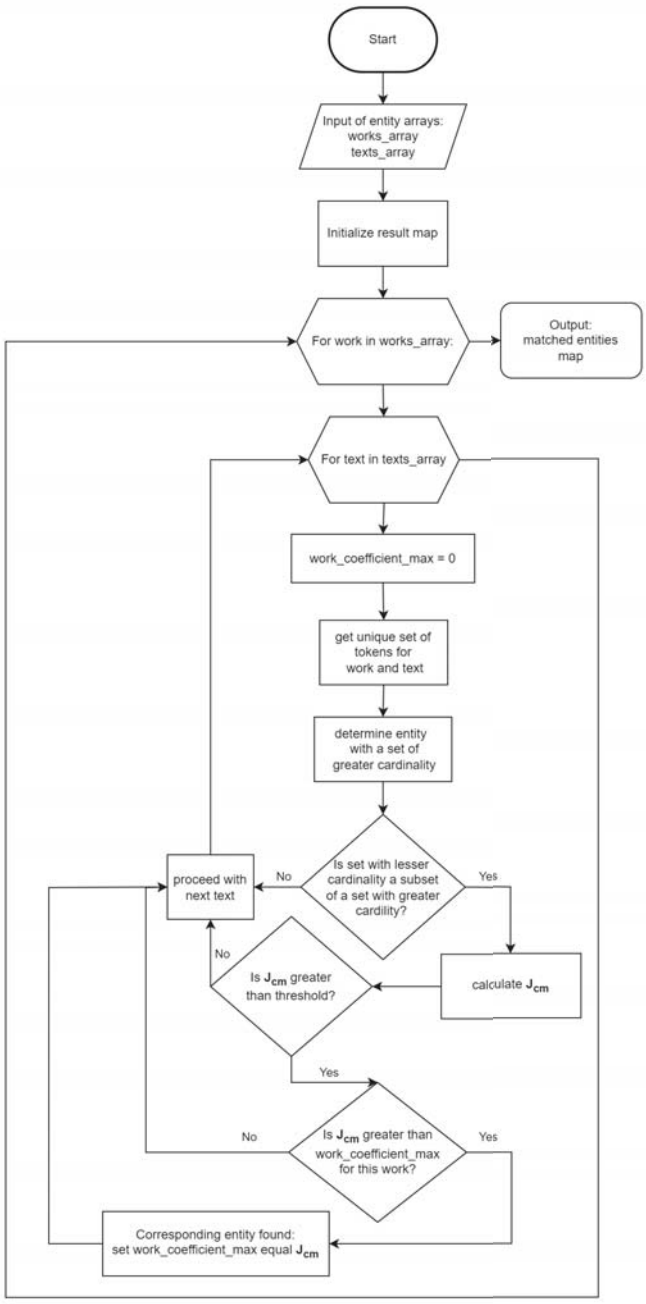

Fig. 6. Scheme of entities matching algorithm

those writing systems by comparison algorithms may require an entirely different approach, i.e. comparison algorithms that rely on phonetic similarity of words' pronunciation in sounds rather letter comparison and this suggestion is one of multiple possible approaches of tackling this aspect of a problem.

The future work will be concentrated on improvements of the method. One of the possible improvements may be the allocation of service parts of speech (for example, particles) and ignorance of them during text characteristics comparison. Similar effect can be achieved by using a pre-trained language model on a large corpus of texts, which is often available for most languages of the world. Specifically for the Russian language, the Natasha model can be used [25]. After implementing this improvement, it is expected that the method will follow its original idea more precisely and demonstrate even better results in the future. The correlation of these entities will allow our research group to move on to the next stage of research – the construction of a knowledge graph of the subject area in relation to the poet.

ACKNOWLEDGMENT

This work is supported by the State Research FFZF-2023-0001.

REFERENCES

[1] N. M. Salih and K. Jacksi, "State of the art document clustering algorithms based on semantic similarity," *Jurnal Informatika*, vol. 14, no. 2, pp. 58–75, 2020.

[2] A. S. Pushkin, *Pushkin's complete works [Polnoe sobranie sochinenii], 1837—1937: V 16 t.* Moscow, Leningrad: Izd-vo AN SSSR, 1937—1959 (In Russ.).

[3] M. N. Virolainen, O. E. Karpeeva, E. O. Larionova, O. C. Muravieva, V. D. Rak, and I. S. Chistova, Eds., *Pushkinskaya encyclopedia. Proizvedenya (A-D) (In Russ.)*. Nestor-Istoria, 2009, vol. 1.

[4] E. I. Vozhik, A. Y. Balakin, G. N. Belyak, E. O. Kazakova, and K. A. Maslinsky, "Index of works and letters of pushkin a. s." 2023. [Online]. Available: https://doi.org/10.31860/openlit-2023.6-B015

[5] K. E. O. Vozhik E. I. and L. R. A., "Corpus of poems by A. S. Pushkin," 2023. [Online]. Available: https://doi.org/10.31860/openlit-2023.8-C005

[6] M. Han, X. Zhang, X. Yuan, J. Jiang, W. Yun, and C. Gao, "A survey on the techniques, applications, and performance of short text semantic similarity," *Concurrency and Computation: Practice and Experience*, vol. 33, no. 5, p. e5971, 2021.

[7] P. A. Hall and G. R. Dowling, "Approximate string matching," *ACM computing surveys (CSUR)*, vol. 12, no. 4, pp. 381–402, 1980.

[8] M. S. Waterman, T. F. Smith, and W. A. Beyer, "Some biological sequence metrics," *Advances in Mathematics*, vol. 20, no. 3, pp. 367–387, 1976.

[9] J. Yang, Y. Li, C. Gao, and Y. Zhang, "Measuring the short text similarity based on semantic and syntactic information," *Future Generation Computer Systems*, vol. 114, pp. 169–180, 2021.

[10] T. Hofmann, "Probabilistic latent semantic analysis," *arXiv preprint arXiv:1301.6705*, 2013.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[12] K. Lund and C. Burgess, "Producing high-dimensional semantic spaces from lexical co-occurrence," *Behavior research methods, instruments, & computers*, vol. 28, no. 2, pp. 203–208, 1996.

[13] P. D. Turney, "Mining the web for synonyms: Pmi-ir versus lsa on toefl," in *European conference on machine learning*. Springer, 2001, pp. 491–502.

[14] R. L. Cilibrasi and P. M. Vitanyi, "The google similarity distance," *IEEE Transactions on knowledge and data engineering*, vol. 19, no. 3, pp. 370–383, 2007.

[15] I. Matveeva, G. Levow, A. Farahat, and C. Royer, "Generalized latent semantic analysis for term representation," in *Proc. of RANLP*, 2005, p. 149.

[16] A. Islam and D. Inkpen, "Second order co-occurrence pmi for determining the semantic similarity of words." in *LREC*, 2006, pp. 1033–1038.

[17] J. Qiang, P. Chen, T. Wang, and X. Wu, "Topic modeling over short texts by incorporating word embeddings," in *Advances in Knowledge Discovery and Data Mining: 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part II 21*. Springer, 2017, pp. 363–374.

[18] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "A latent semantic model with convolutional-pooling structure for information retrieval," in *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, 2014, pp. 101–110.

[19] A. Graves and A. Graves, *Supervised sequence labelling*. Springer, 2012.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[22] S. I. Wang and C. D. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2012, pp. 90–94.

[23] J. Wang and Y. Dong, "Measurement of text similarity: a survey," *Information*, vol. 11, no. 9, p. 421, 2020.

[24] G. Navarro, "A guided tour to approximate string matching," *ACM computing surveys (CSUR)*, vol. 33, no. 1, pp. 31–88, 2001.

[25] "GitHub - natasha: Tools for Russian NLP: segmentation, embeddings, morphology, lemmatization, syntax, NER, fact extraction." [Online]. Available: https://github.com/natasha