Zbornik 25. mednarodne multikonference

INFORMACIJSKA DRUŽBA

Zvezek C

Proceedings of the 25th International Multiconference

INFORMATION SOCIETY

Volume C

2022

Odkrivanje znanja in podatkovna skladišča - SiKDD

Data Mining and Data Warehouses - SiKDD

Urednika • Editors:

Dunja Mladenić, Marko Grobelnik

Ljubljana, Slovenija
10. oktober

10 October
Ljubljana, Slovenia

http://is.ijs.si

Zbornik 25. mednarodne multikonference

INFORMACIJSKA DRUŽBA – IS 2022

Zvezek C

Proceedings of the 25th International Multiconference

INFORMATION SOCIETY – IS 2022

Volume C

Odkrivanje znanja in podatkovna skladišča - SiKDD
Data Mining and Data Warehouses - SiKDD

Urednika / Editors

Dunja Mladenić, Marko Grobelnik

http://is.ijs.si

10. oktober 2022 / 10 October 2022
Ljubljana, Slovenija

Urednika:

Dunja Mladenić,
Department for Artificial Intelligence
Jožef Stefan Institute, Ljubljana

Marko Grobelnik
Department for Artificial Intelligence
Jožef Stefan Institute, Ljubljana

Založnik: Institut »Jožef Stefan«, Ljubljana
Priprava zbornika: Mitja Lasič, Vesna Lasič, Lana Zemljak
Oblikovanje naslovnice: Vesna Lasič

Dostop do e-publikacije:
http://library.ijs.si/Stacks/Proceedings/InformationSociety

Ljubljana, oktober 2022

Informacijska družba
ISSN 2630-371X

Katažni zapis o publikaciji (CIP) pripravili v Narodni in univerzitetni
knjižnici v Ljubljani
COBISS.SI-ID 127444483
ISBN 978-961-264-243-3 (PDF)

PREDGOVOR MULTIKONFERENCI INFORMACIJSKA DRUŽBA 2022

Petindvajseta multikonferenca *Informacijska družba* je preživela probleme zaradi korone. Zahvala za skoraj normalno delovanje konference gre predvsem tistim predsednikom konferenc, ki so kljub prvi pandemiji modernega sveta pogumno obdržali visok strokovni nivo.

Pandemija v letih 2020 do danes skoraj v ničemer ni omejila neverjetne rasti IKTja, informacijske družbe, umetne inteligence in znanosti nasploh, ampak nasprotno – rast znanja, računalništva in umetne inteligence se nadaljuje z že kar običajno nesluteno hitrostjo. Po drugi strani se nadaljuje razpadanje družbenih vrednot ter tragična vojna v Ukrajini, ki lahko pljuskne v Evropo. Se pa zavedanje večine ljudi, da je potrebno podpreti stroko, krepi. Konec koncev je v 2022 v veljavo stopil not raziskovalni zakon, ki bo izboljšal razmere, predvsem leto za letom povečeval sredstva za znanost.

Letos smo v multikonferenco povezali enajst odličnih neodvisnih konferenc, med njimi »Legende računalništva«, s katero postavljamo nov mehanizem promocije informacijske družbe. IS 2022 zajema okoli 200 predstavitev, povzetkov in referatov v okviru samostojnih konferenc in delavnic ter 400 obiskovalcev. Prireditev so spremljale okrogle mize in razprave ter posebni dogodki, kot je svečana podelitev nagrad. Izbrani prispevki bodo izšli tudi v posebni številki revije Informatica (http://www.informatica.si/), ki se ponaša s 46-letno tradicijo odlične znanstvene revije. Multikonferenco Informacijska družba 2022 sestavljajo naslednje samostojne konference:

- Slovenska konferenca o umetni inteligenci
- Izkopavanje znanja in podatkovna skladišča
- Demografske in družinske analize
- Kognitivna znanost
- Kognitonika
- Legende računalništva
- Vseprisotne zdravstvene storitve in pametni senzorji
- Mednarodna konferenca o prenosu tehnologij
- Vzgoja in izobraževanje v informacijski družbi
- Študentska konferenca o računalniškem raziskovanju
- Matcos 2022

Soorganizatorji in podporniki konference so različne raziskovalne institucije in združenja, med njimi ACM Slovenija, SLAIS, DKZ in druga slovenska nacionalna akademija, Inženirska akademija Slovenije (IAS). V imenu organizatorjev konference se zahvaljujemo združenjem in institucijam, še posebej pa udeležencem za njihove dragocene prispevke in priložnost, da z nami delijo svoje izkušnje o informacijski družbi. Zahvaljujemo se tudi recenzentom za njihovo pomoč pri recenziranju.

S podelitvijo nagrad, še posebej z nagrado Michie-Turing, se avtonomna stroka s področja opredeli do najbolj izstopajočih dosežkov. Nagrado Michie-Turing za izjemen življenjski prispevek k razvoju in promociji informacijske družbe je prejel prof. dr. Jadran Lenarčič. Priznanje za dosežek leta pripada ekipi NIJZ za portal zVEM. »Informacijsko limono« za najmanj primerno informacijsko potezo je prejela cenzura na socialnih omrežjih, »informacijsko jagodo« kot najboljšo potezo pa nova elektronska osebna izkaznica. Čestitke nagrajencem!

Mojca Ciglarič, predsednik programskega odbora
Matjaž Gams, predsednik organizacijskega odbora

FOREWORD - INFORMATION SOCIETY 2022

The 25th *Information Society Multiconference* (http://is.ijs.si) survived the COVID-19 problems. The multiconference survived due to the conference chairs who bravely decided to continue with their conferences despite the first pandemics in the modern era.

The COVID-19 pandemic from 2020 till now did not decrease the growth of ICT, information society, artificial intelligence and science overall, quite on the contrary – the progress of computers, knowledge and artificial intelligence continued with the fascinating growth rate. However, the downfall of societal norms and progress seems to slowly but surely continue along with the tragical war in Ukraine. On the other hand, the awareness of the majority, that science and development are the only perspective for prosperous future, substantially grows. In 2020, a new law regulating Slovenian research was accepted promoting increase of funding year by year.

The Multiconference is running parallel sessions with 200 presentations of scientific papers at eleven conferences, many round tables, workshops and award ceremonies, and 400 attendees. Among the conferences, "Legends of computing" introduce the "Hall of fame" concept for computer science and informatics. Selected papers will be published in the Informatica journal with its 46-years tradition of excellent research publishing.

The Information Society 2022 Multiconference consists of the following conferences:

- Slovenian Conference on Artificial Intelligence
- Data Mining and Data Warehouses
- Cognitive Science
- Demographic and family analyses
- Cognitonics
- Legends of computing
- Pervasive health and smart sensing
- International technology transfer conference
- Education in information society
- Student computer science research conference 2022
- Matcos 2022

The multiconference is co-organized and supported by several major research institutions and societies, among them ACM Slovenia, i.e. the Slovenian chapter of the ACM, SLAIS, DKZ and the second national academy, the Slovenian Engineering Academy. In the name of the conference organizers, we thank all the societies and institutions, and particularly all the participants for their valuable contribution and their interest in this event, and the reviewers for their thorough reviews.

The award for life-long outstanding contributions is presented in memory of Donald Michie and Alan Turing. The Michie-Turing award was given to Prof. Dr. Jadran Lenarčič for his life-long outstanding contribution to the development and promotion of information society in our country. In addition, the yearly recognition for current achievements was awarded to NIJZ for the zVEM platform. The information lemon goes to the censorship on social networks. The information strawberry as the best information service last year went to the electronic identity card. Congratulations!

Mojca Ciglarič, Programme Committee Chair
Matjaž Gams, Organizing Committee Chair

KONFERENČNI ODBORI
CONFERENCE COMMITTEES

KAZALO / TABLE OF CONTENTS

Zbornik 25. mednarodne multikonference

INFORMACIJSKA DRUŽBA – IS 2022

Zvezek C

Proceedings of the 25th International Multiconference

INFORMATION SOCIETY – IS 2022

Volume C

Odkrivanje znanja in podatkovna skladišča - SiKDD
Data Mining and Data Warehouses - SiKDD

Urednika / Editors

Dunja Mladenić, Marko Grobelnik

http://is.ijs.si

10. oktober 2022 / 10 October 2022
Ljubljana, Slovenija

PREDGOVOR

Tehnologije, ki se ukvarjajo s podatki so v devetdesetih letih močno napredovale. Iz prve faze, kjer je šlo predvsem za shranjevanje podatkov in kako do njih učinkovito dostopati, se je razvila industrija za izdelavo orodij za delo s podatkovnimi bazami, prišlo je do standardizacije procesov, povpraševalnih jezikov itd. Ko shranjevanje podatkov ni bil več poseben problem, se je pojavila potreba po bolj urejenih podatkovnih bazah, ki bi služile ne le transakcijskem procesiranju ampak tudi analitskim vpogledom v podatke – pojavilo se je t.i. skladiščenje podatkov (data warehousing), ki je postalo standarden del informacijskih sistemov v podjetjih. Paradigma OLAP (On-Line-Analytical-Processing) zahteva od uporabnika, da še vedno sam postavlja sistemu vprašanja in dobiva nanje odgovore in na vizualen način preverja in išče izstopajoče situacije. Ker seveda to ni vedno mogoče, se je pojavila potreba po avtomatski analizi podatkov oz. z drugimi besedami to, da sistem sam pove, kaj bi utegnilo biti zanimivo za uporabnika – to prinašajo tehnike odkrivanja znanja v podatkih (data mining), ki iz obstoječih podatkov skušajo pridobiti novo znanje in tako uporabniku nudijo novo razumevanje dogajanj zajetih v podatkih. Slovenska KDD konferenca pokriva vsebine, ki se ukvarjajo z analizo podatkov in odkrivanjem znanja v podatkih: pristope, orodja, probleme in rešitve.

FOREWORD

Data driven technologies have significantly progressed after mid 90's. The first phases were mainly focused on storing and efficiently accessing the data, resulted in the development of industry tools for managing large databases, related standards, supporting querying languages, etc. After the initial period, when the data storage was not a primary problem anymore, the development progressed towards analytical functionalities on how to extract added value from the data; i.e., databases started supporting not only transactions but also analytical processing of the data. At this point, data warehousing with On-Line-Analytical-Processing entered as a usual part of a company's information system portfolio, requiring from the user to set well defined questions about the aggregated views to the data. Data Mining is a technology developed after year 2000, offering automatic data analysis trying to obtain new discoveries from the existing data and enabling a user new insights in the data. In this respect, the Slovenian KDD conference (SiKDD) covers a broad area including Statistical Data Analysis, Data, Text and Multimedia Mining, Semantic Technologies, Link Detection and Link Analysis, Social Network Analysis, Data Warehouses.

PROGRAMSKI ODBOR / PROGRAMME COMMITTEE

Emotion Recognition in Text using Graph Similarity Criteria

Nadezhda Komarova, Inna Novalija, Marko Grobelnik
Jožef Stefan Institute
Jamova cesta 39, Ljubljana, Slovenia
nadezhdakomarova7@gmail.com

ABSTRACT

In this paper, a method of classifying text into several emotion categories employing different measures of similarity of two graphs is proposed. The emotions utilized are happiness, sadness, fear, surprise, anger and disgust, with the latter two joined into one category. The method is based on representing a text as a graph of n-grams; the results presented in the paper are obtained using the value of 5 for n: the n-grams were the sequences of 5 characters. The graph representation of the text was constructed based on observing which n-grams occur close together in the text; additionally, frequencies of their connections were utilized to assign edge weights. To classify the text, the graph was compared with several emotion category graphs based on different graph similarity criteria. The former relate to common vertices, edges, and the maximum common subgraphs. The evaluation of the model on the test data set shows that utilizing the construction of the maximum common subgraph to obtain the graph similarity measure results in more accurate predictions. Additionally, employing the number of common edges as a graph similarity criterion yielded more accurate results compared to employing the number of common vertices to measure the similarity between the two graphs.

KEYWORDS

emotion recognition, text classification, machine learning, graphs, graph similarity

1 INTRODUCTION

Emotion recognition is a problem that can be connected to different fields such as natural language processing, computer vision, deep learning, etc. [4] In this paper, the focus is on the task of recognizing emotions in texts.

In the literature, several approaches have been introduced that target this problem. Some of them employ vertex embedding vectors for emotion detection and recognition from text. The embedding vectors grasp the information related to semantics and syntax; however, a limitation of such approaches is that they do not capture the emotional relationship that exists between words. Some methods attempting to alleviate this issue include building a neural network architecture adopting pre-trained word representations. [3] Some text classification approaches employ n-grams to construct the text representation, e.g., to deal with the task of language identification. [9]

In this paper, the approach to emotion recognition employs n-grams to obtain graph representation of text. The text is viewed as a sequence of characters that is divided into n-grams, i.e., shorter overlapping sequences of characters as presented in Figure 1.

In Section 2, it is further explained how the graph of n-grams is constructed for a given text and how an emotion label is assigned to the text based on the similarity with the emotion category graphs. Afterwards, in Section 3, the method is compared with related approaches.

In Section 4, an overview of results is focused on differences between the performance of the model when different graph similarity criteria are used. It is followed by the discussion of the model's limitations in Section 5.

2 PROPOSED METHOD

2.1 Constructing the Graph of n-grams

The method used in the paper to obtain text representation in the form of the graph of n-grams is the following.

- The given text was separated into n-grams of characters. Also, different values of n have been tested. The results in Section 4, use $n = 5$. The n-grams into which the given text was split were overlapping.
- The n-grams obtained in this way were utilized to represent the labels of vertices of the graph.
- The edges of the graph were created in the following manner. The ends of edges were the vertices that corresponded to n-grams that occurred close to each other in the text, e.g., the edge is connecting the first n-gram at the beginning of the text with the second n-gram (these two n-grams would overlap with each other), as seen in Figure 1. Different values have been tested for the maximal distance between the two vertices allowed for these two vertices to still be connected with the edge. The results in Section 4, use the value of 7.
- Performance of the model with both, the directed and the undirected graphs, has been tested.

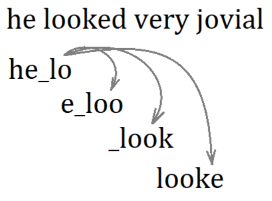


Figure 1: Constructing the edges between the 5-grams that occur close to each other

In Figure 2, it is depicted how the edges are constructed between the vertices labelled with n-grams. For the clarity of representation, each n-gram is shown connected to 3 other n-grams instead of 7. It is important to note that if the same n-grams occurred in the text more than once, there was still only one vertex with this n-gram as a label: the connections of the n-gram have been aggregated at a single vertex.

Additionally, the graph constructed is weighted. The weights of the edges are obtained utilizing the frequencies of connections of n-grams in the given text. In other words, the edge weights are initialized to 0. Then, when constructing the graph of n-grams for a text, every time a certain edge would be added, instead of adding it, the weight of the edge is increased by 1.

Afterwards, the edge weights are normalized to be in the range (0, 1); hence, the edge weights are more comparable among the graphs of n-grams for different texts.

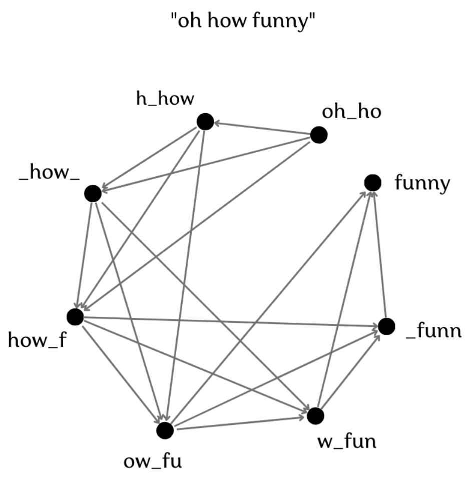


Figure 2: Constructing the edges between the 5-grams in the text fragment *"oh how funny"*

2.2 Constructing the Emotion Category Graphs

The core of the method is the construction of the graph of n-grams as described in Section 2.1. In the data set used to tune the model, there were shorter texts labelled with one of the following 5 emotions: happy, sad, surprised, fearful, or angry-disgusted. Overall, there were 1207 sentences included in the data set; out of this, the model was trained using 1086 sentences (to construct the emotion category graphs) and evaluated on 121 sentences (the split proportion is 90 : 10).

The process of obtaining the emotion category graphs is presented below.

(1) The data set was split into 5 parts containing only the text labelled with the same emotion.
(2) Then, the texts in each part of the data set were used to obtain 5 graphs corresponding to each emotion.
   (a) This process can be viewed as for each text labelled with a certain emotion, constructing the graph of n-grams as explained in Section 2.1.
   (b) Afterwards, merge these graphs separately for different emotions to obtain 5 larger graphs of n-grams; during the merging process, the edges are aggregated in such a way that there are not any two vertices in the emotion category graph sharing the same label (the character n-gram to which they correspond).

2.3 Assigning an Emotion to a Given Text

Utilizing the 5 emotion category graphs corresponding to different emotions, for a given text, it is determined, to which emotion the text most likely corresponds. For that, the pairwise similarity measures of the graph of the given text and of the 5 emotion category graphs are employed.

In other words, it is tested, to which of the 5 graphs the graph of the given text is most similar and the corresponding emotion is assigned to the given text.

Several similarity criteria of the two graphs have been explored.

(1) The number of *vertices* common to both graphs: the vertices are considered common if they share the same label (the n-gram they represent) in both graphs.
(2) The number of *edges* common to both graphs: the edge is considered common if the same vertices (vertices with the same labels) are the endpoints of the edge in both graphs and the edge weights are the same.
(3) The number of vertices in the *maximum common subgraph* (MCS) of the two graphs. Finding the maximum common subgraph is equivalent to finding a graph with the maximum number of vertices so that it is a subgraph of each of the two graphs. [8]
(4) The number of edges in the maximum common subgraph (MCS) of the two graphs.
(5) $z = \frac{m(m-1)}{2} - e$, where m denotes the number of vertices in the maximum common subgraph of the two graphs, and e denotes the number of edges in the maximum common subgraph.

3 RELATED WORK

In the literature describing related approaches to text classification and emotion recognition, deep learning models are often utilized to obtain high-quality predictions. [7]

Apart from the approaches that employ word embedding vectors [6], there are also methods that connect neural networks and graphs. Such approaches may be similar to the method described in this paper since the graph representation of text may be obtained in a similar way based on the semantic connections between words. One example of this kind of model is the graph neural network that is enhanced by utilizing BERT to obtain semantic features. [11]

The crucial part of the method in this paper is the graph similarity criterion that is used when comparing the graph of the given text with different emotion category graphs. The similar way as the construction of the maximum common subgraph is used in this method, it can be employed in combination with the probabilistic classifiers. [10]

The approach in this paper, on the other hand, does not employ probabilistic classifiers such as Bayes Classification or Support Vector Machine. [2] Instead, the emotion for which the similarity measure between the corresponding emotion category graph and the graph of the given text is maximised is assigned to the text.

Additionally, it is important to note that it is possible to incorporate alternative graph similarity criteria, e.g., related to subgraph matching, edit distance, belief propagation, etc. [5]

4 RESULTS

4.1 Experimental Setup

The data set used to train and evaluate the model was the one distributed by Cecilia Ovesdotter Alm. [1] It included the sentences each labelled with one of the following emotions: happiness, sadness, fear, surprise, anger, and disgust. The latter two emotions were joined into one category.

During the evaluation stage, for each sentence, a corresponding emotion was predicted, e.g., the text *"then the servant was*

Table 1: Results of text classification using directed graphs

Similarity criterion	Accuracy	Precision	Recall	F1
Common vertices	0.488	0.506	0.332	0.323
Common edges	0.537	**0.683**	0.408	0.432
z	0.372	0.074	0.200	0.108
Vertices in the MCS	0.570	0.622	0.426	0.446
Edges in the MCS	**0.579**	0.625	**0.454**	**0.478**

Table 2: Results of text classification using undirected graphs

Similarity criterion	Accuracy	Precision	Recall	F1
Common vertices	0.488	0.506	0.332	0.323
Common edges	0.554	**0.669**	0.429	**0.460**
z	0.372	0.074	0.200	0.108
Vertices in the MCS	0.545	0.527	0.399	0.406
Edges in the MCS	**0.570**	0.581	**0.439**	0.453

greatly frightened and said it may perhaps be only a cat or a dog" was labelled fearful, while the text *"he looked very jovial did little work and had the more holidays"* was recognized to be related to the emotion of happiness.

The value of n that appeared to yield the best results and was also used to obtain the results in Tables 1 and 2 was 5. Furthermore, each 5-gram (except those at the end of the text) is connected to 7 5-grams further in the text.

In Tables 1 and 2, the "common edges" criterion means that the two edges from both graphs are considered common if they have the same weight and the same endpoints.

Additionally, in Table 1, z denotes the difference between the the actual number of edges in the maximum common subgraph and the number of edges in the complete graph with m vertices, where m is the number of vertices in the maximum common subraph.

In the trials that yielded the results in Table 1, the edges were directed and in the trials that yielded the results in Table 2, the edges were undirected.

4.2 Analysis

From the results in Table 1 and 2, it may be noticed that the highest accuracy on the test data set was achieved when the number of edges in the maximum common subgraph was used as the similarity measure. In Table 1, the second highest accuracy was achieved when the number of vertices in the maximum common subgraph was utilized.

From this, it may be observed that the construction of the maximum common subgraph reflects the similarity better in certain cases; possible reasons may be that deeper semantic relationships can be captured this way since connections between multiple n-grams are considered at the same time.

In Tables 3 and 4, the confusion matrices are presented for the trials when the number of edges in the maximum common subgraph was used as the criterion of graph similarity.

From the Tables 1 and 2, it is evident that this similarity criterion corresponded to the highest accuracy of predictions for both undirected and directed graphs. However, the accuracy corresponding to this similarity criterion is higher when the graphs are directed (0.579 compared to 0.570).

Table 3: Confusion matrix: directed graph, number of edges in the MCS as the similarity criterion

Actual/pred.	Happy	Fearful	Surpr.	Sad	Angry-Disg.
Happy	**43**	1	0	0	1
Fearful	7	**6**	1	3	0
Surprised	6	1	**2**	1	1
Sad	12	1	0	**12**	1
Angry-Disg.	11	2	0	2	**7**

Table 4: Confusion matrix: undirected graph, number of edges in the MCS as the similarity criterion

Actual/pred.	Happy	Fearful	Surpr.	Sad	Angry-Disg.
Happy	**42**	1	0	1	1
Fearful	8	**6**	1	2	0
Surprised	6	1	**1**	1	2
Sad	11	1	0	**13**	1
Angry-Disg.	11	2	0	2	**7**

Table 5: Confusion matrix: directed graph, number of common edges as the similarity criterion

Actual/pred.	Happy	Fearful	Surpr.	Sad	Angry-Disg.
Happy	**42**	1	0	2	0
Fearful	10	**4**	0	3	0
Surprised	6	0	**2**	3	0
Sad	13	0	1	**12**	0
Angry-Disg.	16	1	0	0	**5**

Table 6: Confusion matrix: undirected graph, number of common edges as the similarity criterion

Actual/pred.	Happy	Fearful	Surpr.	Sad	Angry-Disg.
Happy	**41**	1	0	2	1
Fearful	11	**4**	0	2	0
Surprised	6	0	**2**	3	0
Sad	12	0	1	**13**	0
Angry-Disg.	14	1	0	0	**7**

Furthermore, the accuracy corresponding to the similarity criterion being the number of the common edges (considering both the endpoints and the weight of the edge) is higher by 0.017 when the graphs are undirected than when the graphs are directed (0.554 compared to 0.537). When the graphs utilized are undirected, the model might be more flexible regarding the exact order of the words that occur together.

In Tables 5 and 6, confusion matrices are presented for the trials when the number of edges common to both graphs, considering the endpoints and the weights of the edges, was used as the the criterion of graph similarity.

5 DISCUSSION

A strength of the approach presented in this paper is the ability to capture the context of the given words on different levels; this is related to the process of constructing the edges of the graph by connecting n-grams that occur together in the text. Additionally, the breadth of the contextual frame considered may be varied by altering the number of *n*-grams with which a certain *n*-gram is connected when constructing the edges.

However, overall, the accuracy values noted in Tables 1 and 2, were not very high possibly indicating that the training data set was not large enough. Moreover, the data set did not include texts corresponding to different emotions in even proportions resulting in an imbalance which could have also had a detrimental influence on the quality of predictions. The confusion matrices (Tables 3, 4, 5, and 6) indicate, e.g., that the texts were often falsely assigned the emotion of happiness since it was the most abundant class in the data set.

One of the limitations of the design of the model described it that although it may be reasonable to expect that to obtain more accurate predictions on the test data set, training the model (obtaining the emotion category graphs) on a larger corpus of texts is needed, this may bring a significant rise in computational complexity since the category graphs would possess significantly larger amounts of vertices and edges.

This is especially important if the maximum common subgraphs are constructed when obtaining a similarity measure, since for each text in the test data set, a maximum common subgraph would have to be constructed several times: between the graph of *n*-grams for a given text and each emotion category graph (5 such graphs in this case).

A possible solution to the problem of having too large category graphs might be reducing the length of *n*-grams, i.e., using smaller values of *n*, and hence reducing the number of vertices in the graph.

Also, reducing the number of *n*-grams with which a certain *n*-gram is connected when constructing the edges of the graph may be investigated as a possible solution. However, if this value is too low, too much contextual information may be lost; therefore, it appears necessary that for each value of n, the optimal number of *n*-grams with which a certain *n*-gram is connected is determined experimentally.

6 CONCLUSION

In this paper, the model that utilizes graph similarity criteria to classify a given text into one of the emotion categories is described. The core of the method is to construct a graph of *n*-grams for a given text and to compare this graph to each of the emotion category graphs. The text is classified into the emotion category, the graph of which yielded the highest similarity value when compared to the graph of the given text.

From the results of the trials noted in Tables 1 and 2, it may be concluded that among the graph similarity criteria described, that number of edges in the maximum common subgraph resulted in the highest quality of predictions.

Furthermore, it may also be noted that employing the number of edges common to both graphs resulted in higher prediction accuracy than using the number of common vertices (0.537 and 0.488 accuracy for the directed graphs).

This may appear to be intuitively reasonable as using edges may seem to incorporate more contextual information. Additionally, it may be important to investigate the effect of the difference between the size of the graph of *n*-gram for the given text and the size of the emotion category graph on the probability that the same connections between the two *n*-grams are found in both graphs. Moreover, it may be more probable that the same vertices (vertices labelled with the same *n*-gram) are contained in both graphs resulting in more noisy data.

To conclude, the future work on the task of emotion recognition related to the proposed method may, on the one hand, be focused on employing alternative graph similarity measures in addition to those described in this paper, e.g., those connected to deriving the edit distance or to the belief propagation. [5] Furthermore, clustering algorithms may be used to obtain the patterns characteristic to the emotion categories and further employ them for the emotion recognition task. To this end, both, the vertex clustering algorithms as well as the clustering of graphs as objects, might be utilized. Additionally, graph neural network architecture may be built along with incorporating the graphs of *n*-grams as the input for the network.

7 ACKNOWLEDGEMENTS

This work was supported by the Slovenian Research Agency under the project J2-1736 Causalify and the European Union through Odeuropa EU H2020 project under grant agreement No 101004469.

REFERENCES

[1] Alm, E. C. O. Affect in text and speech, 2008.
[2] Bahritidinov, B., and Sanchez, E. Probabilistic classifiers and statistical dependency: The case for grade prediction. pp. 394–403.
[3] Batbaatar, E., Li, M., and Ryu, K. H. Semantic-emotion neural network for emotion recognition from text. *IEEE Access 7* (2019), 111866–111878.
[4] Guo, J. Deep learning approach to text analysis for human emotion detection from big data. *Journal of Intelligent Systems 31*, 1 (2022), 113–126.
[5] Koutra, D., Ramdas, A., Parikh, A., and Xiang, J. Algorithms for graph similarity and subgraph matching, 2011.
[6] Li, S., and Gong, B. Word embedding and text classification based on deep learning methods. *MATEC Web of Conferences 336* (01 2021), 06022.
[7] Prasanna, P., and Rao, D. Text classification using artificial neural networks. *International Journal of Engineering and Technology(UAE) 7* (01 2018), 603–606.
[8] Quer, S., Marcelli, A., and Squillero, G. The maximum common subgraph problem: A parallel and multi-engine approach. *Computation 8*, 2 (may 2020), 48.
[9] Tromp, E., and Pechenizkiy, M. Graph-based n-gram language identification on short texts. *Proceedings of Benelearn 2011* (01 2011), 27–34.
[10] Violos, J., Tserpes, K., Varlamis, I., and Varvarigou, T. Text classification using the n-gram graph representation model over high frequency data streams. *Frontiers in Applied Mathematics and Statistics 4* (2018).
[11] Yang, Y., and Cui, X. Bert-enhanced text graph neural network for classification. *Entropy (Basel) 23* (11 2021).

SLOmet – Slovenian Commonsense Description

Adrian Mladenic Grobelnik
Department for Artificial Intelligence,
Jozef Stefan Institute
Ljubljana Slovenia
adrian.m.grobelnik@ijs.si

Erik Novak
Department for Artificial Intelligence,
Jozef Stefan Institute,
Jozef Stefan International Postrgraduate School
Ljubljana Slovenia
erik.novak@ijs.si

Dunja Mladenic
Department for Artificial Intelligence,
Jozef Stefan Institute,
Ljubljana Slovenia
dunja.mladenic@ijs.si

Marko Grobelnik
Department for Artificial Intelligence,
Jozef Stefan Institute
Ljubljana Slovenia
marko.grobelnik@ijs.si

ABSTRACT

This paper presents Slovenian commonsense description models based on the COMET framework for English. Inspired by MultiCOMETs approach to multilingual commonsense description, we finetune two Slovenian GPT-2 language models. Experimental evaluation based on several performance metrics shows comparable performance to the original COMET GPT-2 model for English.

KEYWORDS

deep learning, commonsense reasoning, multilingual natural language processing, slovenian language model, gpt-2

1 Introduction

Recent research [1] into commonsense representation and reasoning in the field of natural language understanding has demonstrated promising results for automatic commonsense generation. Given a simple sentence or common entity, such technology can generate plausible commonsense descriptions relating to it. However, further testing on complex sentences, uncommon entities, or by increasing the quantity of requested commonsense descriptions usually gives nonsensical results.

Following the recent success on the automatic generation of commonsense descriptions proposed in COMET-ATOMIC 2020 [1], we focus on extending the COMET framework to the Slovenian language. We investigate the impact of different Slovenian language models on the overall performance of commonsense description generation. In our previous research [2], we expanded on an existing approach for automatic knowledge base construction in English [3] to work on different languages. We utilized the original ATOMIC dataset [4]. This was performed by finetuning the original English GPT model from COMET 2019 on automatically translated Slovenian data and evaluated based on exact overlap for the generated commonsense descriptions. Evaluations were performed on a small subset of 100 sentences. In this work we use the updated ATOMIC-2020 dataset [1] and finetune two Slovenian GPT-2 language models. We evaluate the models' performance using several performance metrics including BLEU, CIDEr, METEOR and ROUGE-L. The evaluation is performed on several thousand sentences and entities; we investigate how the predicted commonsense descriptions' performance relates to the language model used. Furthermore, given the complexity of the Slovenian language compared to English, we anticipate a noticeable drop in performance across all metrics for the Slovenian language models.

The main contributions of this paper are (1) the comparison of the performance of commonsense description models using different Slovenian language models and the English model, (2) a comprehensive evaluation using a variety of performance metrics. An additional contribution (3) is the Slovene ATOMIC-2020 dataset acquired by machine translation from the original English dataset [6].

The rest of this paper is organized as follows: Section 2 provides the data description. Section 3 describes the problem and the experimental setting. Section 4 exhibits our evaluation results. The paper concludes with discussion and directions for future work in Section 5.

2 Data Description

To train the Slovenian commonsense description models, we use data from the ATOMIC-2020 dataset, as proposed in the COMET framework for English. The ATOMIC-2020 dataset consists of English sentences and entities, labelled by up to 23 commonsense relation types describing their semantics.

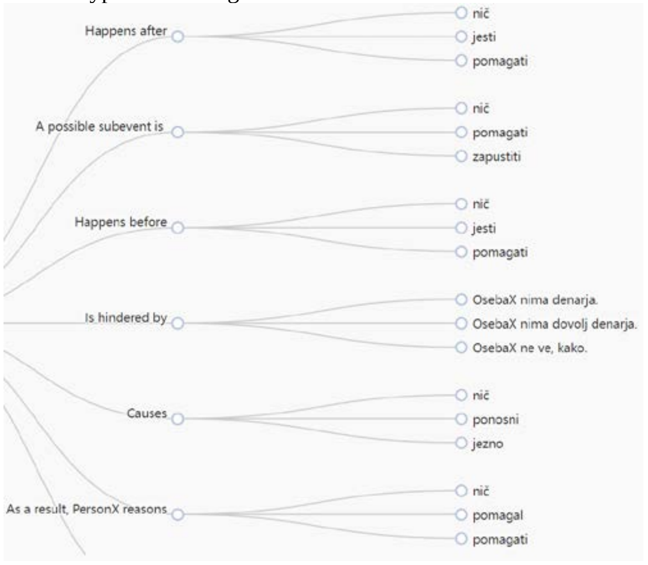


Figure 1 Close-up of "Event-Centered" descriptor values predicted for an example Slovene sentence "PersonX is sad" ("OsebaX je žalostna" in Slovenian)

We refer to them as descriptors, 9 of which are identical to those used in our previous research [2]. The 23 descriptors are organized into 3 categories: “Physical-Entity”, “Event-Centered”, and “Social-Interaction”. The “Physical-Entity” descriptors capture knowledge about the usage, location, content, and other properties of objects. The “Event-Centered” descriptors include IsAfter, Causes and other descriptors describing events. The “Social-Interaction” descriptors include xIntent, xNeed, oReact to distinguish between causes and effects in social settings. An example of a part of a labeled sentence is shown in Figure 1.

Sentences and entities were manually labelled by human workers on Amazon Turk; they were assigned open-text values for 23 commonsense descriptors, reflecting the workers' subjective commonsense knowledge. For instance, when workers were given the sentence “PersonX chases the rabbit” and asked to label it for the “xWant” descriptor, one wrote “catch the rabbit” and another wrote “cook the rabbit for dinner”. A more detailed explanation can be found in the ATOMIC-2020 paper. There are 1.33 million (possibly repeating) descriptor values. The distribution of data across the descriptors is depicted in [1].

To finetune our Slovenian language models, we have automatically translated the sentences, entities, and descriptor values from the ATOMIC-2020 dataset from English to Slovenian. The translation was done using DeepL’s Translate API [7]. We have found that while the majority of inspected translations were of good quality, there were also incorrect translations due to word disambiguation problems. Nevertheless, we conclude that the dataset is of good enough quality to be used for our experiments. The translated dataset is publicly available [6].

3 Problem Description and Experimental Setting

The addressed problem is predicting the most likely values for each descriptor in the Slovene-translated ATOMIC-2020 dataset, given a Slovenian input sentence or entity. We take inspiration from the approach proposed in MultiCOMET [2].

To compare the performance of the models, we utilize a variety of performance metrics described below. Each performance metric is a value between 0 and 1 indicating the quality of a generated commonsense descriptor value. Values closer to 1 represent higher quality descriptions.

BLEU — Bilingual Evaluation Understudy was first used to evaluate the quality of machine translated text by examining the overlap of candidate text n-grams in the reference text. BLEU-1 only uses 1-grams in the evaluation, while BLEU-4 only considers 4-grams. [8]

CIDEr — Consensus-based Image Description Evaluation was originally used to measure image description quality. It first transforms all n-grams to their root form, then calculates the average cosine similarity between the candidate and reference TF-IDF vectors. [9]

METEOR — Metric for Evaluation of Translation with Explicit Ordering is a metric initially used for evaluating machine translation input. The metric is based on the harmonic mean of unigram precision and recall with other features such as stemming and synonymy matching. [10]

ROUGE-L — Recall-Oriented Understudy for Gisting Evaluation is a metric used for evaluating machine produced summaries or translations against a set of human-produced references. The score is calculated using Longest Common Subsequence based statistics, which involves finding the longest subsequence common to all sequences in a set. [11]

Comparison of the Slovene commonsense models was performed by finetuning two state-of-the-art Slovene GPT-2 language models: macedonizer/sl-gpt2 [12], gpt-janez [13]. As a reference model, we used the original COMET-2020 GPT2-XL English language model [1]. Moving forward, we will refer to our Slovenian finetuned models as “COMET sl-gpt2” and “COMET gpt-janez”.

4 Experimental Results

We performed a train, test, and development split on the ATOMIC-2020 dataset identical to the split used in COMET-2020. Our evaluation split consisted of over 150,000 descriptor values with their corresponding sentences and entities.

We finetuned our Slovene commonsense models on our training set consisting of over 1 million descriptor values. Both models were trained for 3 epochs under the same parameters; the maximum input length was set to 50, the maximum output length was set to 80; the training was performed using a train batch size of 64. The model updates were performed using the weighted adam optimizer [14] with the starting learning rate set to 10^{-5}. The experiment’s implementation can be found on our GitHub repository [5].

Model	Language	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	METEOR	ROUGE-L
COMET sl-gpt2	Slovene	0.297	0.150	0.086	0.058	0.487	0.207	0.383
COMET gpt-janez	Slovene	**0.324**	**0.174**	**0.108**	**0.076**	**0.508**	**0.225**	**0.397**
COMET (GPT2-XL)	English	0.407	0.248	0.171	0.124	0.653	0.292	0.485

Table 1: Comparison of the two Slovene commonsense models with the English model at the bottom.

Experimental results show performance comparable to the original COMET-2020 English model. Both Slovene models were comparable to the English model across all metrics, “COMET gpt-janez” consistently outperformed “COMET sl-gpt2” achieving a METEOR score of 0.225 compared to 0.207. The performance gap was smallest for BLEU-4, as all models struggled to produce generations whose 4-grams overlapped with those in the reference set. The gap in performance between the Slovene and English models could be attributed to multiple factors. The English model from COMET-2020 was trained for longer on more capable hardware and is larger. Moreover, the machine translation done to acquire our dataset can be erroneous at times.

To illustrate the performance of the models, we investigate their generated descriptor values on the same inputs. Table 2 shows a side-by-side example comparison of the descriptor values generated by our three models, given the same input sentence in their respective language. Table 3 compares the models on an example entity. For the example sentence “Marko went to the shop”, the descriptor “oWant” indicates what the others want as a result of the event. “COMET gpt-janez” generates a valid output “None” but fails to provide alternatives. The other two models agree on the most likely descriptor value being “None” (“nič” in Slovenian) and provide plausible alternatives. The “IsBefore” descriptor relates to possible events following the input event. In our case, “COMET gpt-janez” gives the most plausible output of “Buys something”. The other two models provide still plausible outputs including “Is in the pet store” and “PersonX buys a new car”.

Marko je šel v trgovino (Marko went to the shop)

Descriptor	COMET sl-gpt2	COMET gpt-janez	COMET (GPT2-XL)
oWant	Nič	Nič	None
	Se zahvaliti osebiX	Nič	To give him a receipt
	se zahvaliti	Nič	To give him a discount
IsBefore	Zaslužiti denar	Kupiti nekaj	PersonX buys a new car
	V trgovino za hišne ljubljenčke	Kupiti nekaj	PersonX takes the car back home
	V trgovino z živili	Kupiti nekaj	PersonX buys a new one

Table 2: Illustrative example comparing the output of the three models on the same input sentence across two descriptors.

For our example entity “car”, the descriptor “ObjectUse” describes possible usages for that entity. Table 3 shows all models are capable of generating plausible descriptor values for such common entities. Nevertheless, the descriptor “HasProperty” proves challenging for the Slovenian models, suggesting a car is “crazy” and is “found in the car”. The English model gives reasonable outputs such as “Found in parking lot”.

Avto (car)

Descriptor	COMET sl-gpt2	COMET gpt-janez	COMET (GPT2-XL)
ObjectUse	Vožnja do trgovine	Priti do hiše	Drive to the store
	Vožnja do hiše	Priti do hiše	Get to the store
	Vožnja do cilja	Priti do hiše	Drive to the restaurant
HasProperty	Noro	Najden v avtomobilu	Found in parking lot
	Vrata	Najden v avtomobilu	Found on road
	Pohištvo	Najden v avtomobilu	Found in car dealership

Table 3: Illustrative example comparing the output of the three models on the same input entity across two descriptors.

In our example sentence and entity, COMET gpt-janez returns the same output when different commonsense descriptors are requested. We have observed this for all input sentences and entities thus far. We presume such results are due to the trained parameters in the original gpt-janez model, as macedonizer/sl-gpt2 was finetuned using the same workflow and returns different descriptor values. While unsure of the exact cause, we reason it could be due to an insufficient vocabulary or unoptimized choice of parameters during training.

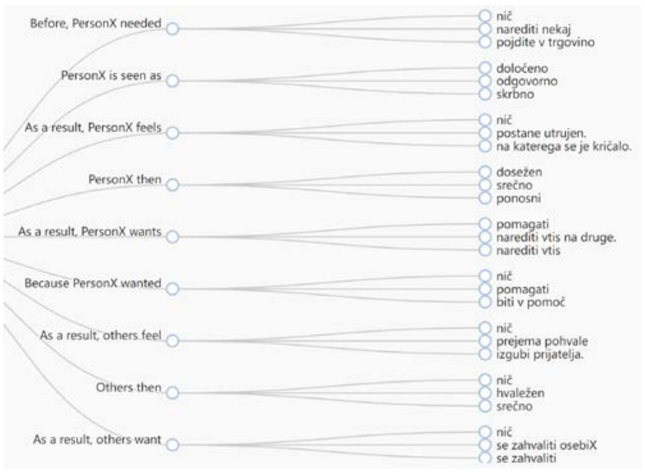


Figure 2 Close-up of “Social-Interaction” descriptor values predicted for an example Slovene sentence “John is very important” (“Janez je zelo pomemben” in Slovenian)

Figures 1, 2 and 3 show the outputs generated by “COMET sl-gpt2” for three different inputs. Figure 2 visualizes the output for the sentence “John is very important”. Outputs include “PersonX is then accomplished, happy, proud” and “As a result, others want none, to thank PersonX”. We can see that for many descriptors the highest ranked output is “None” (“nič” in Slovenian), indicating no commonsense inference can be made.

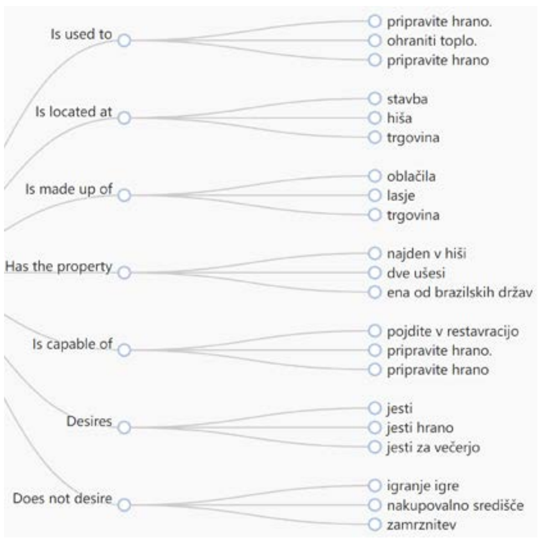


Figure 3 Close-up of "Physical-Entity" descriptor values predicted for an example Slovene entity "banana"

Figure 3 exhibits the output for the entity "banana", the model claims the banana can be used to prepare food, is located in a building or shop, desires to be eaten for dinner and does not desire to be frozen. On the other hand, the model claims the banana is made up of clothes and is capable of going to a restaurant. This is likely due to the overall significantly lower number of physical-entity descriptor values provided in the ATOMIC-2020 dataset.

In Figure 1 we can see the "Event-Centered" descriptors for the sentence "PersonX is sad". Top descriptor values are again "None", but the model also claims it is more difficult for PersonX to be sad, if PersonX has no money.

5 Discussion

This paper applied an existing approach to multilingual commonsense description to the Slovene language. To implement our approach, we machine translated the ATOMIC-2020 dataset to Slovene and finetuned two Slovene commonsense models. We compared our models to the original English commonsense model from COMET-2020 and achieved comparable experimental results across multiple performance metrics. Among others, our models achieved a 0.487 CIDEr score, a 0.383 ROUGE-L score, and a BLEU-1 score of 0.297.

Through examination of individual examples, we observed that while "COMET gpt-janez" has the highest performance scores on the Slovene language, it fails to provide alternative descriptor values. "COMET sl-gpt" provides multiple values for the same descriptor, but in average has lower performance. It is important to emphasize the models were trained on subjective commonsense knowledge provided by individual humans. For example, workers labelled the sentence "PersonX digs holes" with the descriptor values "PersonX plants a garden" and "PersonX places fence posts in the holes" for the "IsBefore" descriptor. While both labels are plausible for some context, they are not necessarily true.

Possible directions for future work include evaluating the models' performance for individual descriptors, as there are drastic differences in quantity of training data and lengths of values across them. After achieving results comparable to the original English commonsense model COMET-2020 GPT2-XL, we intend to finetune and evaluate models for other languages.

ACKNOWLEDGMENTS

The research described in this paper was supported by the Slovenian research agency under the project J2-1736 Causalify, the RSDO project funded by the Development of Slovene in a Digital Environment project, and the Humane AI Net European Unions Horizon 2020 project under grant agreement No 952026.

REFERENCES

[1] Hwang, J.D., Bhagavatula, C., Le Bras, R., Da, J., Sakaguchi, K., Bosselut, A., & Choi, Y. (2021). COMET-ATOMIC 2020: On Symbolic and Neural Commonsense Knowledge Graphs. AAAI.
[2] Mladenic Grobelnik, A., Mladenić, D., & Grobelnik, M. (2020). MultiCOMET - Multilingual Commonsense Description. In Proc. SiKDD 2020, Ljubljana, Slovenia (pp. 37–40).
[3] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, Yejin Choi. (2019). COMET: Commonsense Transformers for Automatic Knowledge Graph Construction.
[4] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, Yejin Choi. (2019). ATOMIC: An Atlas of Machine Commonsense for If-Then Reasoning. Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, USA. Allen Institute for Artificial Intelligence, Seattle, USA.
[5] SLOmet-ATOMIC 2020 Github https://github.com/eriknovak/RSDO-SLOmet-atomic-2020#slomet-atomic-2020-on-symbolic-and-neural-commonsense-knowledge-graphs-in-slovenian-language Accessed 30.08.2022
[6] ATOMIC-2020 Slovene Machine Translated Data https://www.dropbox.com/sh/gs8iqcwpwkaqkuf/AAAmnCqG89JOz_umtq42MMxxa?dl=0 Accessed 30.08.2022
[7] DeepL Translate API https://www.deepl.com/pro-api Accessed 30.08.2022
[8] Papineni, Kishore & Roukos, Salim & Ward, Todd & Zhu, Wei Jing. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation.
[9] Vedantam, R., Lawrence Zitnick, C., & Parikh, D. (2015). Cider: Consensus-based image description evaluation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4566-4575).
[10] Lavie, Alon & Denkowski, Michael. (2009). The METEOR metric for automatic evaluation of Machine Translation. Machine Translation. 23. 105-115.
[11] Lin, C.-Y. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In Text Summarization Branches Out.
[12] Documentation page for "macedonizer/sl-gpt2" on HuggingFace https://huggingface.co/macedonizer/sl-gpt2 Accessed 1.09.2022
[13] gpt-janez supporting project: RSDO https://www.cjvt.si/rsdo/en/project/ Accessed 30.08.2022
[14] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, in: International Conference on Learning Representations, 201

Measuring the Similarity of Song Artists using Topic Modelling

Erik Calcina
Jožef Stefan Institute
Jamova cesta 39
Ljubljana, Slovenia

Erik Novak
Jožef Stefan International Postgraduate School
Jožef Stefan Institute
Jamova cesta 39
Ljubljana, Slovenia

ABSTRACT

In music streaming platforms, it is necessary a recommendation system to provide users with similar songs of what they already listen and also recommend new artists they might be interested in. In this paper, we present a method to find similarities between artists that uses topic modelling. We have evaluated the method using a data set with music artists and their lyrics. The results show the method finds similar artists, but also is dependant on the quality of the generated topic clusters.

KEYWORDS

song lyrics, topic modelling, clustering, sentence embeddings, language models

1 INTRODUCTION

Nowadays, there are a plenty of music platforms to choose from and listen to music. There, new artists appear every day and many different songs are published. If we take into account all that have been created, we get a large selection of songs which can increase the difficulty of finding suitable songs or artists to listen to.

To find a suitable artist or songs, different aspects can be considered. One such aspect can be the topic of the song; a song topic can be interpreted as the main subject of the song, for example it can be an emotion, an event, a message, or something else. When searching for suitable artists one could decide to search for artists who have songs on similar topics.

In this paper, we propose an topic modeling-based approach for measuring the similarity of the music artists based only on their song lyrics. The approach uses language models for generating song embeddings used to create the topic clusters. These topic clusters are then analyzed to find the similar artists. The experiment was performed on a data set of songs corresponding to fourteen (14) music artists. While the experiment shows that similar artists can be detected using the approach, there is still room for improving its performance.

The main contribution of this paper is a novel approach for detecting similar music artists using topic modelling.

The reminder of the paper is structured as follows: Section 2 contains the overview of the related work on using topic modelling on song data sets. Next, we present the methodology in Section 3, and describe the experiment setting in Section 4. The experiment results are found in Section 5, followed by a discussion in Section 6. Finally, we conclude the paper and provide ideas for future work in Section 7.

2 RELATED WORK

Related works to our topic modeling approach use Latent Dirichlet Allocation (LDA) [1]. One work uses a topic modeling technique for sentiment classification, classifying between happy and sad songs, by using generated topics created with LDA and Heuristic Dirichlet Process [12]. From a data set consisting of 150 lyric they've been able to retrieve the sub-division of two defined sentiment classes [3]. Another work used LDA and Pachinko allocation [7] on a large data set for assessing the quality of the generated topics with applying supervised topic modeling approach. [8]. In our paper we use topic modeling to generate a set of topic clusters used to calculate the similarity between artists.

3 METHODOLOGY

In this section, we present the methodology used in this paper. We present the topic modeling approach used to generate the topic clusters, followed by a description of how the topic clusters are used to measure the similarity between the artists.

3.1 Topic Modeling

To create the topic clusters we use BERTopic [5], a method which uses document embeddings with clustering algorithms to create topic clusters. While BERTopic is described in a separate work, we present a brief description of its workflow. The workflow is also presented in Figure 1.

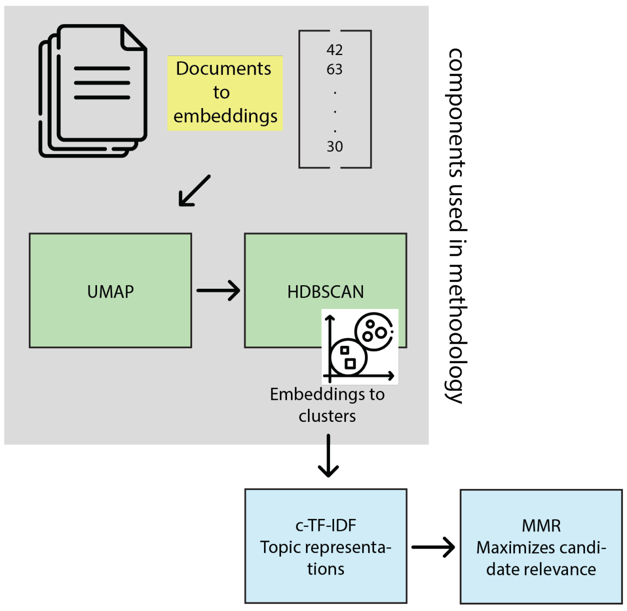

Documents to embeddings
42
63
.
.
.
30
components used in methodology
UMAP
HDBSCAN
Embeddings to clusters
c-TF-IDF Topic representations
MMR Maximizes candidate relevance

Figure 1: The BERTopic methodology workflow. The highlighted part is used in our approach. The image has been designed using resources from Flaticon.com.

Document Embeddings. Document vector representations are generated using a sentence-transformer [11] model. The model creates a semantic representation of the documents, which allows measuring the semantic similarity. The available models support creation of both monolingual and multilingual vectors. Since the embeddings will be used as an input of a clustering algorithm, dimensionality reduction is performed to improve the clustering results. The dimensionality reduction algorithm used is UMAP [10].

Document Clustering. Once the document embeddings are prepared, they are input into a clustering algorithm to create the topic clusters. The algorithm used is HDBSCAN [9], an optimized extension of the DBSCAN [4] algorithm. The chosen algorithm creates clusters based on the density of the document embedding space, which allows the documents to not be assigned to a cluster if it's not similar to any of the neighbouring documents.

Topic Word Description. Once the topic clusters are created, a topic word description is generated using the document's text. For each cluster the TF-IDF score is calculated for each word found in any of the cluster's documents; the scores are called cluster TF-IDF (c-TF-IDF). The words with the highest c-TF-IDF score are then chosen as the topic word description. Furthermore, maximal marginal relevance (MMR) is performed to diversify the selected words by measuring both the words relevance to the documents, and its similarity to the other selected words. Note that the topic word description were used only for the preliminary analysis of our work, but not for measuring artists similarity.

3.2 Artists' Similarity using Topic Clusters

Once the topic clusters are created, the similarity between artists can be measured. First, for each topic we count the songs that corresponds to a particular artist. This gives us the number of songs an artist has in a particular topic. To ensure that the presence is strong enough, we decide to remove the artists from a topic if the number of their associated songs is below some threshold. The threshold is set to five (5) in order to ensure that the songs were not assigned to a cluster by coincidence. Afterwards, for each pair of artists we calculate their similarity using the following equation:

$$\mathrm{sim}(a,b) = \frac{|A \cap B|}{|A|}, \tag{1}$$

where A is the set of topics of artist a, and B is the set of topics of artist b.

4 EXPERIMENT

We now present the experiment setting. First, we introduce the data set used and its pre-processing steps. Next, we describe the implementation details.

4.1 Dataset

To test our approach, we use a dataset with raw lyrics data [2]. The dataset consists of 218,210 rows containing the following attributes:

- *Song name.* The name of the song.
- *Release year.* The year when the song was released.
- *Song artist.* The name of the artist.
- *Artist genre.* The genre of the song.
- *Song lyrics.* The lyrics text of the song.

The attributes used in our analysis are song name, artist and lyrics.

Data Processing. For our experiment we took fourteen (14) artists of various degrees of similarity. This reduces the data set to 4,470 rows which is 2.05% of the whole data set.

After reviewing the lyrics, we realized that the data set has many song variations by the same artist, which can be seen as duplicates. To find and remove the duplicates, we created the TF-IDF representations for the songs, and calculated the cosine similarity with all other songs of the same artist; if the similarity is greater than 50% it was labeled as a duplicate and removed from the data set. This resulted in a smaller data set containing 3,455 song lyrics.

The final data set statistics used for our experiments is shown in Table 1.

Table 1: The experiment data set statistics. For each artist we denote the music genre of the artist (genre), the number of their songs in the data set (songs), and the average number of words in the song's lyrics (avg. length).

Artist	genre	songs	avg. length
black-sabbath	Rock	160	184
bon-jovi	Rock	320	266
dio	Rock	127	203
aerosmith	Rock	208	226
ac-dc	Rock	171	193
coldplay	Rock	138	174
50-cent	Hip-Hop	318	502
2pac	Hip-Hop	259	648
eminem	Hip-Hop	369	640
black-eyed-peas	Hip-Hop	119	463
celine-dion	Pop	182	230
britney-spears	Pop	225	313
frank-sinatra	Jazz	356	133
ella-fitzgerald	Jazz	503	156
Together	-	3,455	319

4.2 Implementation details

In this section, we present the details of how the approach is developed.

Language model. The method uses the pre-trained Sentence Transformer model, more precisely the `all-mpnet-base-v2` model[1], available via the HuggingFace's transformer library [13]. It can take up to 384 tokens as one input, which is more than the average number of words in our data set, and returns a 768 dimensional dense vectors. The vectors have been shown to be appropriate for task such as clustering and semantic search.

Dimensionality reduction. To perform dimensionality reduction, we set the UMAP parameters as follows: Fist, the number of neighboring sample points used when making the manifold approximation is set to five (5), to make the algorithm use the local proximity of the documents. Second, we set the dimensionality of the embeddings to one (1). This values were selected using hyper-parameter tuning.

[1]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Clustering algorithm. In the HDBSCAN algorithm, the minimum number of documents in a cluster is set to five (5).

5 RESULTS

In this section, we present the experiment results. We analyze the topic clusters, followed by the description of the finding on artist's similarity.

Topic Cluster Analysis. The experiment has generates 215 topic clusters, out of which only 107 have at least one artist with more than five (5) songs in it. The cluster containing songs that are deemed as outliers is not included in the analysis.

The statistics of the topic clustering is shown in Table 2. Evidently, artists with a larger number of songs are spread over several topic clusters than those with less songs.

Table 2: Topic clustering results. For each artist we show the number of different topics the artist is asociated with (topics), and the average number of their songs in the associated topics (avg. songs).

Artist	topics	#avg. songs
black-sabbath	6	5
bon-jovi	10	6
dio	4	7
aerosmith	9	6
ac-dc	7	5
coldplay	2	5
50-cent	17	9
2pac	13	9
eminem	18	9
black-eyed-peas	3	12
celine-dion	8	6
britney-spears	12	6
frank-sinatra	16	8
ella-fitzgerald	28	8

Artists' Similarity Analysis. The artists' similarity is shown in Figures 2 and 3, which show the heatmaps of the absolute and relative co-occurrence of artists in topic clusters, respectively.

By looking at rows of Figure 2, we see the number of common topics with other artists. For example, by taking 50-cent with his 17 topics, we see that he shares five (5) of them with 2pac, one (1) with black-eyed-peas, one (1) with ac-dc, and six (6) with eminem. From this we conclude that 50-cent, 2pac and eminem have more topics in common than the rest of the artists. In other words, 50-cent is more similar to the 2pac and eminem than to the rest of the artists.

Figure 3 shows the similarities calculated using Equation 1. The similarities become more visible, but at the same time can be also misleading. Artists with smaller number of topics can result in higher similarity with other artists with higher number of topics. For example, Coldplay have two (2) topics, one of which is shared with Bon Jovi. Despite the fact that only one topic is in common, it is unlikely they have a similarity of 50%.

6 DISCUSSION

In this section we discuss the advantages and disadvantages of the proposed methodology, and its possible improvements.

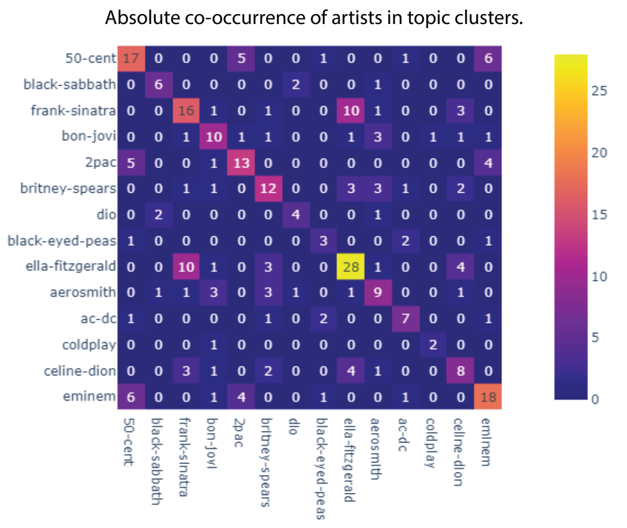


Figure 2: The absolute co-occurrence of artists in topic clusters.

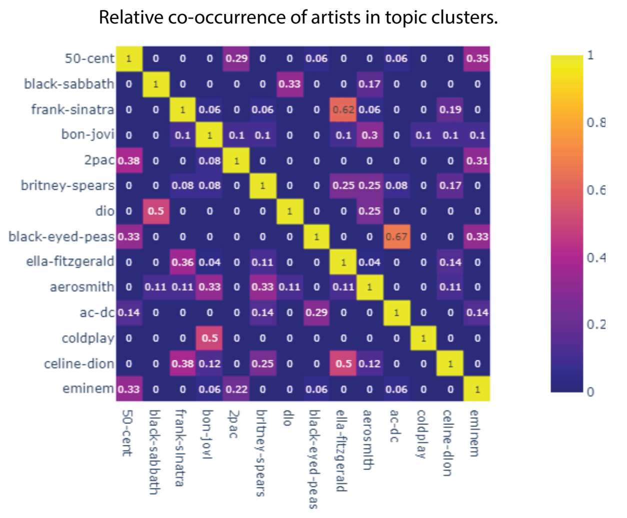


Figure 3: The relative co-occurrence of artists in topic clusters. Artists with smaller number of topics can result in higher similarity with other artists.

Language Models Limitations. The chosen language model `all-mpnet-base-v2` supports a maximum sequence length of 384 tokens which is the downside of this model for our experiment. Although the average number of words in the song lyrics is below the input limit, some artist have songs that are longer than that. However, songs have repeating sections, e.g. chorus, which is most likely inside the first 384 words. Therefore, the language models may not create a representation out of the whole song's lyrics, but it might capture the majority because of the song's repeated text.

Clustering Algorithm Selection. The clustering algorithm HDBSCAN can create a cluster consisting of examples, which do not fall into any of the topic clusters. It is convenient when instead of forcing songs into clusters, it labels them as outliers. The downside is when the majority of songs are labeled as outliers. To avoid this, other clustering algorithms that assign a cluster to every document can be used, for example K-means clustering [6].

6.1 Topic Cluster Discussion

Some artists with a small number of songs have a lower number of topics assigned, which is a problem for finding similarities. On the other side artists with higher number of songs tend to have more topics. Additionally, to avoid taking into account small number of artist co-occurrances, which can be a product of data noise, a filter threshold can be considered to remove them from the final analysis.

7 CONCLUSION

In this paper we present a way to measure similarity between music artists using topic modeling. We cluster lyrics and compare artists based on the generated topic clusters. The results have shown that the approach finds similar artists. However, it is heavily dependent on the number and quality of the topic clusters.

In the future, we intend to apply the methodology on a larger data set of song lyrics and artists. In addition, we intend to use all of the topic cluster information (including topic word descriptions) in order to improve the methodology's performance.

ACKNOWLEDGMENTS

This work was supported by the Slovenian Research Agency and the Slovene AI observatory under proposal no. V2-2146.

REFERENCES

[1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. “Latent dirichlet allocation”. In: *J. Mach. Learn. Res.* 3 (2003), pp. 993–1022. ISSN: 1532-4435. DOI: http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993. URL: http://portal.acm.org/citation.cfm?id=944937.

[2] Connor Brennan, Sayan Paul, Hitesh Yalamanchili, Justin Yum. *Classifying Song Genres Using Raw Lyric Data with Deep Learning.* Accessed August 30, 2022. https://github.com/hiteshyalamanchili/SongGenreClassification. 2018.

[3] Maibam Debina Devi and Navanath Saharia. “Exploiting Topic Modelling to Classify Sentiment from Lyrics”. In: *Machine Learning, Image Processing, Network Security and Data Sciences.* Ed. by Arup Bhattacharjee et al. Singapore: Springer Singapore, 2020, pp. 411–423. ISBN: 978-981-15-6318-8.

[4] Martin Ester et al. “A density-based algorithm for discovering clusters in large spatial databases with noise”. In: AAAI Press, 1996, pp. 226–231.

[5] Maarten Grootendorst. “BERTopic: Neural topic modeling with a class-based TF-IDF procedure”. In: *arXiv preprint arXiv:2203.05794* (2022).

[6] Xin Jin and Jiawei Han. “K-Means Clustering”. In: *Encyclopedia of Machine Learning.* Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2010, pp. 563–564. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_425. URL: https://doi.org/10.1007/978-0-387-30164-8_425.

[7] Wei Li and Andrew McCallum. “Pachinko allocation: DAG-structured mixture models of topic correlations”. In: *ICML '06: Proceedings of the 23rd international conference on Machine learning.* New York, NY, USA: ACM, 2006, pp. 577–584. ISBN: 1595933832. DOI: 10.1145/1143844.1143917. URL: http://portal.acm.org/citation.cfm?id=1143917.

[8] Alen Lukic. *A comparison of topic modeling approaches for a comprehensive corpus of song lyrics.* Tech. rep. Tech report, Language Technologies Institute, School of Computer Science ..., 2015.

[9] Leland McInnes and John Healy. “Accelerated Hierarchical Density Based Clustering”. In: *2017 IEEE International Conference on Data Mining Workshops (ICDMW).* 2017, pp. 33–42. DOI: 10.1109/ICDMW.2017.12.

[10] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction.* 2018. DOI: 10.48550/ARXIV.1802.03426. URL: https://arxiv.org/abs/1802.03426.

[11] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, Nov. 2019. URL: https://arxiv.org/abs/1908.10084.

[12] Chong Wang, John Paisley, and David Blei. “Online variational inference for the hierarchical Dirichlet process”. In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics.* JMLR Workshop and Conference Proceedings. 2011, pp. 752–760.

[13] Thomas Wolf et al. “Transformers: State-of-the-Art Natural Language Processing”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations.* Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. URL: https://aclanthology.org/2020.emnlp-demos.6.

Exploring the Impact of Lexical and Grammatical Features on Automatic Genre Identification

Taja Kuzman
taja.kuzman@ijs.si
Jožef Stefan Institute and Jožef Stefan International Postgraduate School
Jamova cesta 39
Ljubljana, Slovenia

Nikola Ljubešić
nikola.ljubesic@ijs.si
Jožef Stefan Institute
Jamova cesta 39
Ljubljana, Slovenia

ABSTRACT

This study analyses the impact of several types of linguistic features on the task of automatic web genre identification applied to Slovene data. To this end, text classification experiments with the fastText models were performed on 6 feature sets: original lexical representation, preprocessed text, lemmas, part-of-speech tags, morphosyntactic descriptors, and syntactic dependencies, produced with the CLASSLA pipeline for language processing. Contrary to previous work, our results reveal that the grammatical feature set can be more beneficial than lexical representations for this task, as syntactic dependencies were found to be the most informative for genre identification. Furthermore, it is shown that this approach can provide insight into variation between genres.

KEYWORDS

language processing, linguistic features, automatic genre identification, web genres, Slovene

1 INTRODUCTION

Automatic genre identification (AGI) is a text classification task where the focus is on genres as text categories that are defined based on the conventional function and/or the form of the texts. In text classification tasks, texts are generally given to the machine learning models in form of words or characters that are then further transformed into numeric vectors by using bag-of-words representations, or word embeddings created by training deep neural networks on the surface text. However, recent development of tools for linguistic processing for numerous languages, including Slovene, allows transformation of the original running text into various other sets of features to which further transformation into numeric representations can be applied. By learning on these linguistic sets, we get insight into the importance of features that cannot be analysed separately when given the running text, i.e., word meaning, function of a word, and its relation to other words.

When previous work compared importance of various textual feature sets on the performance of the models in automatic genre identification, lexical features, i.e., word or character n-grams, mainly provided the best results ([6], [7]). However, it was noted that by learning on lexical features, the models could learn to classify texts based on the topic instead of genre characteristics, and would not be able to generalize beyond the dataset.

As learning on lexical features can introduce bias towards topic, Laippala et al. (2021) recently experimented with combining lexical with grammatical features, which are represented as part-of-speech tags, conveying information on the word type (e.g., noun, verb). This showed to yield better results than using solely lexical features, and provided more stable models, i.e., models that are able to generalize beyond the training data. Furthermore, their analysis revealed that the importance of feature sets varies between genre categories, and that while some are most efficiently identified when learning on lexical features, others benefit more from grammatical representations.

However, these experiments were in past mostly performed on English datasets. This article is the first to analyse the impact of various feature sets on automatic genre identification applied to Slovene data. This research was made possible by the recent development of the first Slovene dataset, manually annotated with genre, as well as the creation of state-of-the-art language processing tools for Slovene. To compare textual representations, additional feature sets were created from a selection of texts annotated with genre, presented in Section 2, by using common preprocessing methods and language processing (see Section 3). Thus, in this paper, 6 textual representations are compared: 1) original, running text that we consider as our baseline, 2) preprocessed text, i.e. lowercase text without punctuation, digits and stopwords, 3) lemmas, i.e. base dictionary forms of words, 4) part-of-speech (PoS) tags, i.e. main syntactic word types (e.g., noun, verb), 5) morphosyntactic descriptors (MSD), i.e. extended PoS tags which include information on morphosyntactic features (e.g., number, case), 6) syntactic dependencies, i.e. types of dependency relations between words (e.g. subject, object). The feature sets are compared based on their impact on the performance of the fastText models on the automatic text classification task. The results of the experiments, presented in Section 4, give insights into the role of linguistic feature sets on this task and the differences in performance between genre categories.

2 DATASET

For performing experiments in automatic genre identification, the Slovene Web genre identification corpus GINCO 1.0 [2] was used. The dataset consists of the “suitable” subset, annotated with genre, and the “not suitable” subset that comprises texts which can be deemed as noise in the web corpora, e.g., texts without full sentences, very short texts, machine translation etc. In this research, only the “suitable” subset, containing 1002 texts, was used.

The GINCO schema consists of 24 genre labels. However, previous experiments, performed with the fastText model on the entire dataset, showed that the model is not potent enough to differentiate between a large number of labels that are mostly represented by less than 100 texts, reaching micro and macro

Table 1: The original GINCO categories (left) included in the reduced set, and the reduced set of labels (right), used in the experiments, with the total number of texts (later divided between the train, dev and test split) in the parentheses.

GINCO	Reduced Set
News/Reporting Opinionated News	News (198)
Information/Explanation Research Article	Information/Explanation (127)
Opinion/Argumentation Review	Opinion/Argumentation (124)
Promotion Promotion of a Product Promotion of Services Invitation	Promotion (191)
Forum	Forum (48)

F1 scores of 0.352 and 0.217 respectively (see [3]). Therefore, to be able to infer any meaningful conclusions, this article focuses only on the most frequent genre labels, created by merging some labels. Instances of less frequent labels that could not be merged, namely *Instruction*, *Legal/Regulation*, *Recipe*, *Announcement*, *Correspondence*, *Call*, *Interview*, *Prose*, *Lyrical*, *Drama/Script*, *FAQ*, and the labels *Other* and *List of Summaries/Excerpts*, which can be considered as noise, were not used. To focus only on the instances that are representative of their genre labels, texts that were manually annotated as hard to identify (parameter `hard`) were not used in the experiments. Furthermore, paragraphs that were deemed to be noise in the text, e.g., cookie consent text, and were marked by the annotators with the `keep` parameter set to `False`, were left out of the final texts.

Thus, the final set of labels, used in the experiments, shown in Table 1, consists of 5 genre categories, *Information/Explanation*, *News*, *Opinion/Argumentation*, *Promotion* and *Forum*. As shown in the Table, the dataset is imbalanced, with *News* and *Promotion* being the most frequent classes, consisting of almost 200 instances, while *Forum* is the least represented class, consisting of about 50 texts. The subset, consisting of 688 texts in total, followed the original stratified split of 60:20:20, encoded in the GINCO 1.0 dataset, and the models were trained on the training set, tested on the test set, while the dev split was used for evaluating the hyperparameter optimisation.

3 FEATURE ENGINEERING

Feature engineering is a process of identifying features that are most useful for a specific task with the goal of improving performance of a machine learning model. In text classification experiments, basic preprocessing methods are often used to reduce the number of unique lexical features (words or characters) without losing much information which could provide better results. To test whether preprocessing the text improves the results for this task, the first additional feature set was created by preprocessing the running text as extracted from the GINCO dataset. Preprocessing consisted of the following steps: converting text to lowercase, and removing digits, punctuation and function words known as stopwords, e.g., conjunctions, prepositions etc.

In addition to this, various linguistic representations were created by applying linguistic processing to the texts, and replacing words with corresponding lemmas or grammatical tags. The language processing was performed with the CLASSLA pipeline [5]. The following text representations were produced: lexical feature set, consisting of lemmas, and three grammatical feature sets: part-of-speech (PoS) tags, morphosyntactic descriptors (MSD), and syntactic dependencies. The realisation of the created feature sets is illustrated on an example sentence in Table 2.

4 MACHINE LEARNING EXPERIMENTS

4.1 Experimental Setup

The experiments were performed with the linear fastText [1] model which enables text classification and word embeddings generation. The model is a shallow neural network with one hidden layer where the word embeddings are created and averaged into a text representation which is fed into a linear classifier. The model takes as an input a text file where each line contains a separate text instance, consisting of a label and the corresponding document. Thus, for each feature set, appropriate train, test and dev files were created, and the model was trained on each representation separately [1]. To observe the dispersion of results, five runs of training were performed for each feature set. To measure the model's performance on the instance and the label level, the micro and macro F1 scores were used as evaluation metrics.

The hyperparameter search was performed by training the model on the training split of the baseline text and evaluating it on the dev split. The automatic hyperparameter optimisation provided by the fastText model did not yield satisfying results, as three runs of automatic hyperparameter optimisation produced very different results in terms of proposed optimal hyperparameter values and yielded micro F1 0.479 ± 0.02 and macro F1 0.382 ± 0.06. Therefore, we continued searching for optimal hyperparameters by manually changing one hyperparameter at a time

[1]The code for data preparation and machine learning experiments is published here: https://github.com/TajaKuzman/Text-Representations-in-FastText.

Table 2: An example of the feature sets used in the experiments.

Feature Set	Example
Baseline - Running Text	V Laškem se bo v nedeljo, 21.4.2013 odvijal prvi dobrodelni tek Veselih nogic.
Preprocessed Baseline	laškem nedeljo odvijal dobrodelni tek veselih nogic
Lemmas	v Laško se biti v nedelja , 21.4.2013 odvijati prvi dobrodelen tek vesel nogica .
PoS	ADP PROPN PRON AUX ADP NOUN PUNCT NUM VERB ADJ ADJ NOUN ADJ NOUN PUNCT
MSD	Sl Npnsl Px——y Va-f3s-n Sa Ncfsa Z Mdc Vmpp-sm Mlomsn Agpmsny Ncmsn Agpfpg Ncfpg Z
Dependencies	case nmod expl aux case obl punct nummod root amod amod nsubj amod nmod punct

and conducting classification experiments. The optimum number of epochs revealed to be 350, the learning rate was set to 0.7, and the number of words in n-grams to 1. For the other hyperparameters, the default values were used. Manual hyperparameter search revealed to be considerably more effective than automatic optimisation, as it yielded the average micro and macro F1 scores of 0.625 ± 0.004 and 0.618 ± 0.003 respectively, which is in average 0.15 points better micro F1 and 0.24 points better macro F1 compared to the results of automatic optimisation.

To analyse whether our choice of technology is the most appropriate one, we compared the performance of the fastText model, which uses the hyperparameters mentioned above, with the performance of various non-neural classifiers, commonly used in text classification tasks: dummy majority classifier which predicts the most frequent class to every instance, support vector machine (SVM), decision tree classifier, logistic regression classifier, random forest classifier, and Naive Bayes classifier. We used the default parameters for the classifiers. The models are compared based on their performance on the baseline text which was transformed into the TF-IDF representation where necessary. As shown in Table 3, fastText outperforms all other classifiers with a noticeable difference especially in the macro F1 scores, reaching 17 points higher scores than the next best classifier, the Naive Bayes classifier.

Table 3: Micro and macro F1 scores obtained by various classifiers, trained and tested on the baseline text.

Classifier	Micro F1	Macro F1
Dummy Classifier	0.24	0.08
Support Vector Machine	0.49	0.33
Decision Tree	0.34	0.35
Logistic Regression	0.52	0.38
Random Forest classifier	0.51	0.41
Naive Bayes classifier	0.54	0.42
FastText	**0.56**	**0.59**

4.2 Results of Learning on Various Linguistic Features

To explore the role of various textual representations on the automatic genre identification of Slovene web texts, we conducted text classification experiments with the fastText models on 6 feature sets:

- three lexical sets: a) baseline text, i.e., the original running text, b) preprocessed baseline text, i.e., baseline text converted to lowercase and without punctuation, digits and function words, c) lemmas, i.e., words reduced to their base dictionary forms;
- three grammatical sets: a) part-of-speech (PoS), i.e., main word types, b) morphosyntactic descriptors (MSD), i.e., extended PoS tags, c) syntactic dependencies, i.e., types of words defined by their relation to other words.

First, by comparing the baseline representation and the preprocessed representation, we aimed to determine whether common preprocessing methods can improve the results in the AGI task. As shown in Table 4, the results reveal that applying preprocessing methods improves the performance, especially on the micro F1 level. Analysis of the F1 scores obtained for each label in Figure 1 reveals that preprocessing especially improves the identification of *Promotion* and *News*. The two labels are the most frequent genre classes in the dataset which explains larger improvement of the micro F1 scores. If we compare the baseline text and the preprocessed text to the third lexical set, i.e., lemmas, the results show that by using lowercase words, reduced to their dictionary base form, the performance is further improved, although only slightly, as can be seen in Table 4.

Table 4: Average micro and macro F1 scores obtained from five runs of training and testing on each representation separately.

Representation	Micro F1	Macro F1
Baseline Text	0.560 ± 0.00	0.589 ± 0.00
Preprocessed Baseline	0.596 ± 0.00	0.597 ± 0.00
Lemmas	0.597 ± 0.01	0.601 ± 0.00
PoS	0.540 ± 0.01	0.547 ± 0.01
MSD	0.563 ± 0.01	0.536 ± 0.02
Dependencies	**0.610** ± 0.00	**0.639** ± 0.00

Secondly, we compared various lexical and grammatical feature sets, obtained with language processing tools. In previous work, which analysed English genre datasets, lexical features yielded better results than grammatical feature sets ([4], [6], [7]). Our results revealed that this conclusion holds also for Slovene when training on part-of-speech tags. Similar conclusion can be made for the extended part-of-speech tags (MSD) which only slightly improve the micro F1 scores compared to the baseline while there is a decrease in the macro F1 scores (see Table 4). However, the third grammatical feature set, consisting of tags for syntactic dependencies, which was not used in previous work, significantly outperformed the baseline text and all other feature sets. As shown in Figure 1, the improvement is especially noticeable for the categories *Forum*, *Opinion/Argumentation* and *News*. By learning on the dependencies instead on lexical features, the model learns from the structure of the sentences in the text, i.e., the syntax, instead of word meanings that can be more related to topic than genre, which could be the reason why this representation was revealed to be the most beneficial for the task.

As in previous work (see [4]), the experiments have revealed a dependence between the text representation and performance on specific genre labels, which is illustrated in Figure 1. The results show that *Promotion* and *Information/Explanation* can be most successfully identified when learning purely on the meaning of the words, i.e., on lemmas. In contrast to that, for identifying *News*, grammatical representations are more useful than lexical ones. Similarly, *Opinion/Argumentation* benefits more from grammatical feature sets than lexical representations, except in case of the MSD tags which significantly decreased the results for this class, yielding F1 scores below 0.3. Interestingly, although *Forum* is the least frequent label, its features seem to be the easiest to identify in the majority of representations. This genre benefits the most from learning on syntactic dependencies tags, which yielded F1 scores of almost 0.9.

5 CONCLUSIONS

In this paper, we have investigated the dependence of automatic genre classification on the lexical and grammatical representation of text. Our experiments, performed on three lexical and three

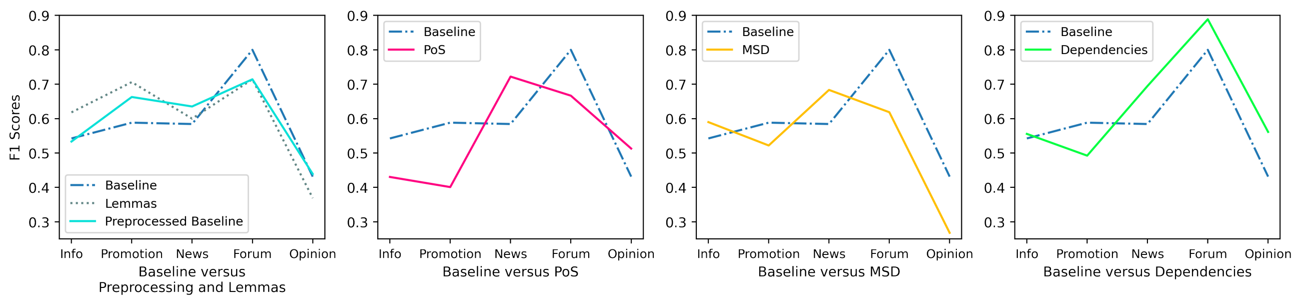


Figure 1: The impact of various linguistic features on the F1 scores of genre labels (*Information/Explanation, Promotion, News, Forum* and *Opinion/Argumentation*).

grammatical feature sets, revealed that the choice of textual representation impacts the results of automatic genre identification. Similarly to previous work, it was revealed that part-of-speech features give worse results than lexical features. However, a grammatical feature set, consisting of syntactic dependencies, that has not been studied in previous work, revealed to be the most beneficial for the automatic genre identification task. Furthermore, the experiments revealed variation between genres regarding the impact of feature sets on the F1 scores of each label. While some genres, such as *Promotion*, benefit more from learning on lexical features, others, such as *Opinion/Argumentation*, benefit more from grammatical representations.

However, it should be noted that this study has been limited to the 5 most frequent genre labels, as the previous experiments showed that the fastText model is not potent enough to identify other categories represented by a small number of instances ([3]). Thus, the results of these experiments give insight into which linguistic features are the most important for differentiating between the five most frequent genres, not for identifying the 24 original labels that encompass all the genre variation found on the web, and include noise. This is why we plan to continue genre annotation campaigns to enlarge the Slovene genre dataset, which would allow extending the analysis to all genre labels. In addition to this, as we are interested in cross-lingual genre identification, in the future, we plan to analyse the importance of linguistic feature sets on the Croatian and English genre datasets to analyse whether the characteristics of genre labels are language independent.

The fastText model was revealed to be useful for the analysis of the impact of linguistic features on the AGI task, however, previous work on automatic genre identification using the GINCO dataset revealed that if the aim of the research is to create the best-performing classifier and not to analyse the impact on the performance, the Transformer-based pre-trained language models are much more suitable for the task ([3]). This was also confirmed by our experiments on the running text, where the base-sized XLM-RoBERTa model reached micro and macro F1 scores 0.816 and 0.813, which is 22–26 points more than the fastText model. Based on the findings from this paper, one of the reasons why the Transformer models perform better could also be that the Transformer text representations incorporate information on syntax as well. In the future, we plan to investigate this further, adapting the classifier heads so that the syntactic information has a larger impact on the classification than the lexical parts of the representation.

ACKNOWLEDGMENTS

This work has received funding from the European Union's Connecting Europe Facility 2014-2020 - CEF Telecom, under Grant Agreement No. INEA/CEF/ICT/A2020/2278341. This communication reflects only the author's view. The Agency is not responsible for any use that may be made of the information it contains. This work was also funded by the Slovenian Research Agency within the Slovenian-Flemish bilateral basic research project "Linguistic landscape of hate speech on social media" (N06-0099 and FWO-G070619N, 2019–2023) and the research programme "Language resources and technologies for Slovene" (P6-0411).

REFERENCES

[1] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

[2] Taja Kuzman, Mojca Brglez, Peter Rupnik, and Nikola Ljubešić. 2021. Slovene web genre identification corpus GINCO 1.0. Slovenian language resource repository CLARIN.SI. (2021). http://hdl.handle.net/11356/1467.

[3] Taja Kuzman, Peter Rupnik, and Nikola Ljubešić. 2022. The GINCO Training Dataset for Web Genre Identification of Documents Out in the Wild. In *Proceedings of the Language Resources and Evaluation Conference*. European Language Resources Association, Marseille, France, 1584–1594. https://aclanthology.org/2022.lrec-1.170.

[4] Veronika Laippala, Jesse Egbert, Douglas Biber, and Aki-Juhani Kyröläinen. 2021. Exploring the role of lexis and grammar for the stable identification of register in an unrestricted corpus of web documents. *Language resources and evaluation*, 1–32.

[5] Nikola Ljubešić and Kaja Dobrovoljc. 2019. What does Neural Bring? Analysing Improvements in Morphosyntactic Annotation and Lemmatisation of Slovenian, Croatian and Serbian. In *Proceedings of the 7th Workshop on Balto-Slavic Natural Language Processing*. Association for Computational Linguistics, Florence, Italy, (August 2019), 29–34. DOI: 10.18653/v1/W19-3704. https://www.aclweb.org/anthology/W19-3704.

[6] Dimitrios Pritsos and Efstathios Stamatatos. 2018. Open set evaluation of web genre identification. *Language Resources and Evaluation*, 52, 4, 949–968.

[7] Serge Sharoff, Zhili Wu, and Katja Markert. 2010. The Web Library of Babel: evaluating genre collections. In *LREC*. Citeseer.

Stylistic features in clustering news reporting: News articles on BREXIT

Abdul Sittar
abdul.sittar@ijs.si
Jožef Stefan Institute and Jožef Stefan Postgraduate School
Jamova cesta 39
Ljubljana, Slovenia

Jason Webber
jason.webber@bl.uk
British Library
London, United Kingdom

Dunja Mladenić
dunja.mladenic@ijs.si
Jožef Stefan Institute and Jožef Stefan Postgraduate School
Jamova cesta 39
Ljubljana, Slovenia

ABSTRACT

We present a comparison of typical bag-of-words features with stylistic features. We group the news articles published from three different regions of the UK namely London, Wales, and Scotland. Hierarchical clustering is performed using typical bag-of-words and stylistic features. We present the performance of 25 stylistic features and compare them with the bag-of-words. Our results show that bag-of-words are better to be used while clustering news reporting at the regional level whereas stylistic features are better to be used while clustering news reporting at the level of news publishers/newspapers.

KEYWORDS

news reporting, topic modeling, stylistic features, clustering

1 INTRODUCTION

The role of content is an essential research topic in news spreading. Media economics scholars especially showed their interest in a variety of content forms since content analysis plays a vital role in individual consumer decisions and political and economic interactions [6]. The content basically refers to the type of language that is used in the news. It is used to convey meaning and it can impact social and psychological constructs such as social relationships, emotions, and social hierarchy [8]. The everyday act of reading the news is such a big area in which small differences in reporting may shape how events are perceived, and ultimately judged and remembered [5].

News reporting across different regions requires methods to find reporting differences. [7] characterize the relationship between the volume of online opioid news reporting and measures differences across different geographic and socio-economic levels. Scholars across disciplines have explored the institutional, organizational, and individual influences that study the quality and quantity of coverage [3].

Features that could classify news reporting across different regions can be adapted to classify the news. A detailed analysis of textual features is performed by [1] where they derived multiple features for creating clusters of news articles along with their comments. These features include terms in the title, terms in the first sentence, terms in the entire article, etc. Multi-view clustering on multi-model data can provide common semantics to improve learning effectiveness. It exploits different levels of features from the raw features, including low-level features, high-level features, and semantic features [16].

Table 1: List of all the stylistic features that are used for clustering.

No.	Feature	No.	Feature
1.	Percentage of Question Sentences	2.	Average Sentence Length
3.	Percentage of Short Sentences	4.	Average Word Length
5.	Percentage of Long Sentences	6.	Percentage of Semicolons
7.	Percentage of Words with Six and More Letters	8.	Percentage of Punctuation marks
9.	Percentage of Words with Two and Three Letters	10.	Percentage of Pronouns
11.	Percentage of Coordinating Conjunctions	12.	Percentage of Prepositions
13.	Percentage of Comma	14.	Percentage of Adverbs
15.	Percentage of Articles	16.	Percentage of Capitals
17.	Percentage of Words with One Syllable	18.	Percentage of Colons
19.	Percentage of Nouns	20.	Percentage of Determiners
21.	Percentage of Verbs	22.	Percentage of Digits
23.	Percentage of Adjectives	24.	Percentage of Full stop
25.	Percentage of Interjections		

The news coverage registers the occurrence of specific events promptly and reflects the different opinions of stakeholders [4]. We take Brexit as an event to be researched on the topic of news reporting differences across the different regions of the UK. On 23 June 2016, the British electorate voted to leave the EU. This event has already been studied following different aspects such as fundamental characteristics of the voting population, driver of the vote, political and social patterns, and possible failures in communication [2, 9]. In this paper, we explore how different stylistic features help in clustering news articles related to Brexit than bag-of-words (BOW).

Following are the main scientific contributions of this paper:

(1) We present a comparison of clustering (using two different textual features: bag-of-words and stylistic features) for news reporting about Brexit in three different regions (London, Scotland, and Wales) of the UK.
(2) We show in our experiments that the bag-of-words are better to be used while clustering news reporting at the regional level whereas stylistic features are better to be used while clustering news reporting at the level of news publishers/newspapers.

2 RELATED WORK

In this section, we review the related literature about topic modelling, and different types of textual features.

2.1 Topic Modelling

Topic modelling is used to infer topics from the collection of text-document. Some techniques used only frequent words whereas

Table 2: Total number of news articles about Brexit published in three different regions (London, Scotland, and Wales).

Regions	Newspapers	News articles	Total
London	bankofengland.co.uk	8	4248
	bbc.com	2209	
	dailymail.co.uk	768	
	Independent.co.uk	191	
	inews.co.uk	52	
	metro.co.uk	1	
	neweconomics.org	1	
	rspb.org.uk	8	
	theguardian.com	1167	
	theneweuropean.co.uk	1	
	thesun.co.uk	235	
	cityam.com	3	
	conservativewomen.uk	1	
	dailypost.co.uk	1	
	ft.com	2	
	mirror.co.uk	9	
	raeng.org.uk	1	
	standard.co.uk	20	
Scotland	news.stv.tv	533	533
Wales	gov.wales	3	280
	nation.wales	122	
	Walesonline.co.uk	156	

some use pooling to generate relevant topics and maintain coherence between topics [14]. Topics are typically represented by a set of keywords. Examples of such algorithms are the Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA) and Probabilistic Latent Semantic Analysis (LSA). Clustering-based topic modelling is another solution.

2.2 Stylistic Features

News reporting differences can be reflected through one's speech, writing, and images etc [10, 12]. A language independent features have been used for different tasks of NLP such as plagiarism detection, author diarization. These features considers the text of documents as a sequence of tokens (i.e. sentences, paragraphs, documents). On the basis of these tokens, various types of statistics could be drawn from any language [13]. Stylistic features represent the writing style of a document and have been used for understanding the author writing styles in the past [10]. We use it to explore the clustering of the news articles based on their reporting differences across different regions. Table 1 shows the list of 25 stylistic features used for the development of our proposed clustering of news articles.

2.3 Bag-of-words

A bag-of-words model is a way of extracting features from text. It is basically a representation of text that describes the occurrence of words within a document. It firstly identifies a vocabulary of known words and then measures the presence of known words. Topic modelling is typically based on the bag-of-words (BOW). The essential idea of the topic model is that a document can be represented by a mixture of latent topics and each topic is a distribution over words [11].

3 DATA COLLECTION

We collected news articles reporting on Brexit in the English language from the UK Web Archive (UKWA). The dataset consists of 5061 news articles after pre-processing. Due to the unavailability of news articles from other regions of the UK, we selected only the regions (London, Scotland, and Wales) which have a sufficient amount of news articles. Table 2 presents the number of news articles published from different regions and by different news publishers.

4 METHODOLOGY

The presented research focuses on clustering news articles. To this end, we experiment clustering with the combination of different features observing their performance. Our methodology consists on four steps and compares the performance of stylistic features and bag-of-words in clustering news articles, as shown in Figure 1.
In the first step, we select Brexit under topic and themes on UK web archive[1]. After crawling the list of news articles, we extracted the meta data of news publishers from Wikipedia-infobox. The meta-data extraction process is explained in our previous work [15]. In this process, we extracted the headquarters of news publishers. Due to the unavailability of news articles from other regions of the UK, we selected only the regions (London, Scotland, and Wales) which have a sufficient amount of news articles.
In the second step, we perform parsing of the html web pages and extract the body text.

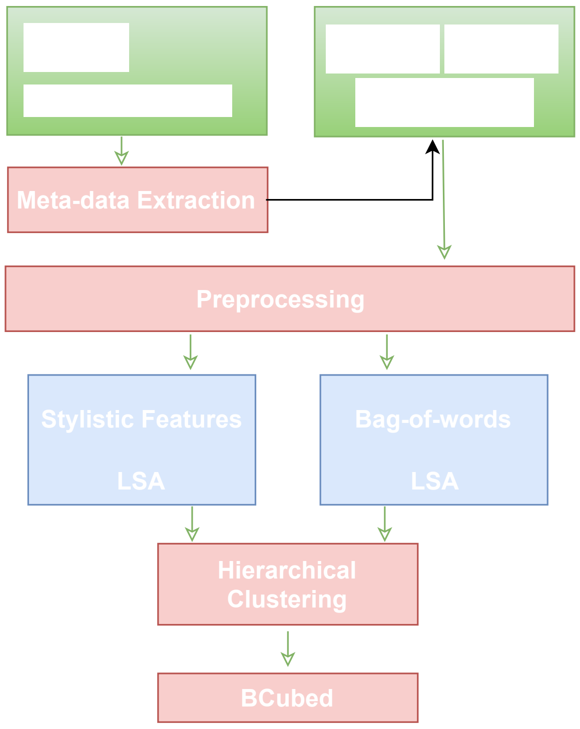


Figure 1: Methodology to clustering regional news using bag-of-words and stylistic features.

[1]https://www.webarchive.org.uk/en/ukwa/collection/910

Since the third step required pre-processing for bag-of-words, we convert the text to lowercase and remove the stop words and punctuation marks. In the third step for the stylistic features, we extract the stylistic features(see Table 1) for all three regions and perform LSA (Latent Semantic Analysis). Similarly, for the bag-of-words, we use the pre-processed text and perform LSA. We also perform LSA on the combination of both types of features. 100 latent dimensions have been used for LSA because it is recommended. We perform LSA and hierarchical clustering using the python library SciPy, and scikit-learn and use the weighted distance between clusters. After performing the LSA, we apply hierarchical clustering and utilize two different types of evaluation measures namely BCubed F1 and Silhouette Scores. For LSA and hierarchical clustering, we use the python library SciPy, and scikit-learn.

5 EXPERIMENTAL EVALUATION

We have performed experimental evaluations using intrinsic (Silhouette) and extrinsic (BCubed-F) evaluation measures. The intrinsic evaluation metrics are used to calculate the goodness of a clustering technique whereas extrinsic evaluation metrics are used to evaluate clustering performance. For extrinsic evaluation, we consider clusters generated by k-means clustering using typical bag-of-words as ground truth clusters. The value of k in k-means clustering ranges from 2 to 20. K-means identifies k centroids, and then allocates every data point to the nearest cluster while keeping the centroids as small as possible. We cannot set the value of k to 1 which means there are no other clusters to allocate the nearest data point.
Silhouette is used to find cohesion. It ranges from -1 to 1. 1 means clusters are well apart from each other and clearly distinguished. 0 means clusters are indifferent, or we can say that the distance between clusters is not significant. -1 means clusters are assigned in the wrong way.
BCubed F-measure defines precision as point precision, namely how many points in the same cluster belong to its class. Similarly, point recall represents how many points from its class appear in its cluster.

- **Silhouette Score:** $S(i) = \frac{b(i)-a(i)}{max(a(i),b(i))}$

where S(i) is the silhouette coefficient of the data point i, a(i) is the average distance between i and all the other data points in the cluster to which i belongs, and b(i) is the average distance from i to all clusters to which i does not belong.

- **BCubed Precision and Recall:**

$$Correctness(i,j) = \begin{cases} 1, if\ L(i) = L(j)\ and\ C(j) = C(j) \\ 0, if otherwise \end{cases}$$

$$BCubed\ Precision = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\epsilon C(i)}\frac{Correctness(i,j)}{|C(i)|}$$

$$BCubed\ Recall = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\epsilon L(i)}\frac{Correctness(i,j)}{|L(i)|}$$

where |C(i)| and |L(i)| denote the sizes of the sets C(i) and L(i), respectively. L(i) and C(i) denote the class and clusters of a point i.

- **BCubed-F Score:** $F = \frac{2\times BcubedPrecision\times BcubedRecall}{BcubedPrecision+BcubedRecall}$

6 RESULTS AND ANALYSIS

Figure 2 shows the three line graphs. Each graph shows Silhouette scores across a different number of clusters (from 2 to 20) representing different regions of the UK such as Scotland, Wales, and London respectively. Blue and red lines represent bag-of-words (BOW) and stylistic features.

We can see that for all three graphs, the silhouette score of stylistic features is significantly high for all three regions except at one point for Scotland. It means that cohesion is higher and the distance between the clusters is more significant using stylistic features than BOW which is mostly too close to 0. It suggests that these features are better at partitioning news articles into clusters than BOW.

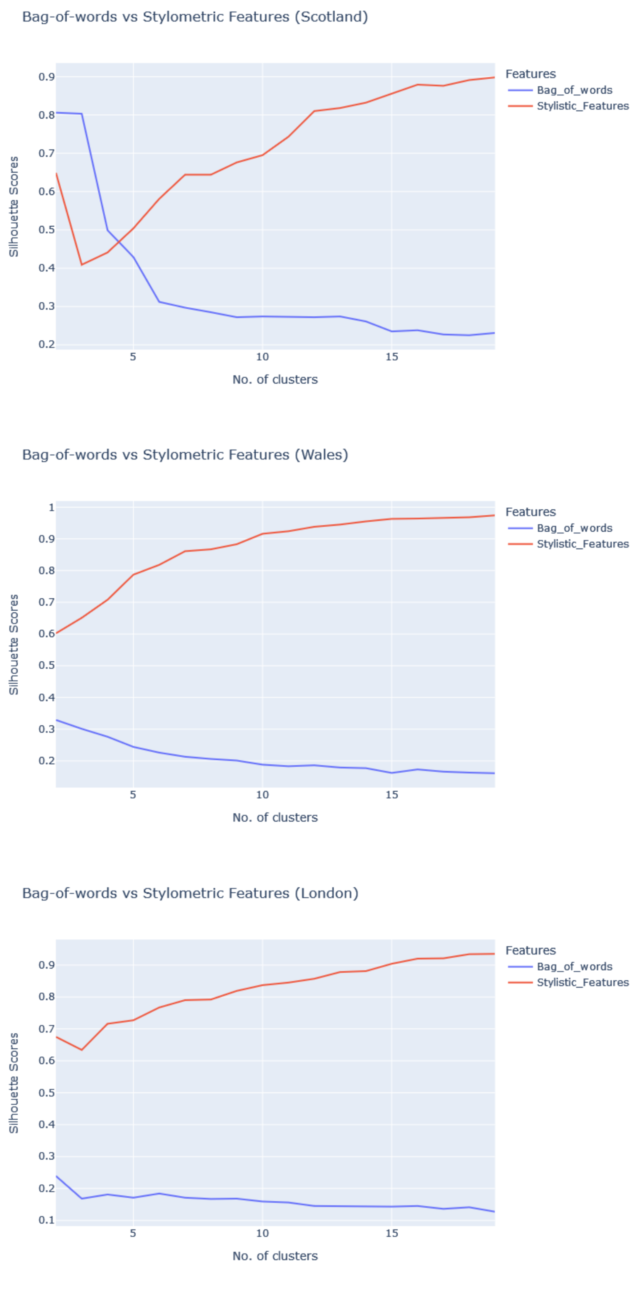


Figure 2: The line graphs represent average silhouette scores across a different number of clusters. The blue line represents the score generated using bag-of-words and the red line represents the score generated using stylistic features. The three-line graphs are generated for three different regions Scotland, Wales, and London respectively.

Table 3: The group of news articles published from three different regions of the UK is considered as ground truth clusters and the Bcubed-F score is calculated using three types of features including bag-of-words, stylistic features, and a combination of both types of features.

No.	Features	Bcubed-F Score
1.	Bag-of-words	0.75
2.	Bag-of-words and stylistic features	0.51
3.	Stylistic features	0.54

Table 4: The group of news articles published from 22 different news publishers of the UK is considered as ground truth clusters and the Bcubed-F score is calculated using three types of features including bag-of-words, stylistic features, and a combination of both types of features.

No.	Features	Bcubed-F Score
1.	Bag-of-words	0.53
2.	Bag-of-words and stylistic features	0.57
3.	Stylistic features	0.66

However, it is insufficient to say that stylistic features are better for news reporting differences at this stage because it is not necessary that the resulting clusters by internal partitioning can be equal to the ones that are based on news reporting differences.

We consider each region (London, Scotland, and Wales) as a ground truth cluster of the news articles published in that region. Table 3 shows Bcubed-F scores when the ground truth clusters were matched with the one that was created using bag-of-words, stylistic features, and a combination of both types of features. Similarly, we consider each newspaper/news publisher shown in Table 2 as a ground truth cluster of the news articles published by that newspaper/news publisher. Table 4 shows Bcubed-F scores when the ground truth clusters were matched with the one that was created using bag-of-words, stylistic features, and a combination of both types of features. The scores using bag-of-words considering regions as ground truth clusters are significantly high (0.75) than stylistic features (0.54) and a combination of all features (0.51). The scores using stylistic features considering newspaper/news publishers as ground truth clusters are significantly high (0.66) than bag-of-words (0.53) and a combination of all features (0.57). The higher scores in regional news reporting suggest that bag-of-words is better to be used for clustering or classification because the newspapers/news publishers report in different styles in a certain region. Similarly, when it comes to classifying or clustering news reporting across different newspapers/news publishers then stylistic features are more useful because the newspapers/news publishers follow a different reporting style.

7 CONCLUSIONS

In this paper, we have presented the comparison of different features observing their performance over clustering news articles. The goal of this work was to investigate the performance of stylistic features and typical bag-of-words. The data consists of news articles about a popular event Brexit that are collected from UKWA. These news articles belong to three different regions of the UK including Scotland, London, and Wales. Our experimental results suggest that bag-of-words are better to be used while clustering news reporting at the regional level whereas stylistic features are better to be used while clustering news reporting at the level of news publishers/newspapers.

ACKNOWLEDGMENTS

The research described in this paper was supported by the Slovenian research agency under the project J2-1736 Causalify and by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 812997.

REFERENCES

[1] Ahmet Aker, Monica Paramita, Emina Kurtic, Adam Funk, Emma Barker, Mark Hepple, and Rob Gaizauskas. 2016. Automatic label generation for news comment clusters. In *Proceedings of the 9th International Natural Language Generation Conference*. Association for Computational Linguistics, 61–69.

[2] Sascha O Becker, Thiemo Fetzer, and Dennis Novy. 2017. Who voted for brexit? a comprehensive district-level analysis. *Economic Policy*, 32, 92, 601–650.

[3] Danielle K Brown and Summer Harlow. 2019. Protests, media coverage, and a hierarchy of social struggle. *The International Journal of Press/Politics*, 24, 4, 508–530.

[4] Honglin Chen, Xia Huang, and Zhiyong Li. 2022. A content analysis of chinese news coverage on covid-19 and tourism. *Current Issues in Tourism*, 25, 2, 198–205.

[5] Elizabeth W Dunn, Moriah Moore, and Brian A Nosek. 2005. The war of the words: how linguistic differences in reporting shape perceptions of terrorism. *Analyses of social issues and public policy*, 5, 1, 67–86.

[6] Frederick G Fico, Stephen Lacy, and Daniel Riffe. 2008. A content analysis guide for media economics scholars. *Journal of Media Economics*, 21, 2, 114–130.

[7] Yulin Hswen, Amanda Zhang, Clark Freifeld, John S Brownstein, et al. 2020. Evaluation of volume of news reporting and opioid-related deaths in the united states: comparative analysis study of geographic and socioeconomic differences. *Journal of Medical Internet Research*, 22, 7, e17693.

[8] Qihao Ji, Arthur A Raney, Sophie H Janicke-Bowles, Katherine R Dale, Mary Beth Oliver, Abigail Reed, Jonmichael Seibert, and Arthur A Raney. 2019. Spreading the good news: analyzing socially shared inspirational news content. *Journalism & Mass Communication Quarterly*, 96, 3, 872–893.

[9] Moya Jones. 2017. Wales and the brexit vote. *Revue Française de Civilisation Britannique. French Journal of British Studies*, 22, XXII-2.

[10] Ifrah Pervaz, Iqra Ameer, Abdul Sittar, and Rao Muhammad Adeel Nawab. 2015. Identification of author personality traits using stylistic features: notebook for pan at clef 2015. In *CLEF (Working Notes)*. Citeseer, 1–7.

[11] Zengchang Qin, Yonghui Cong, and Tao Wan. 2016. Topic modeling of chinese language beyond a bag-of-words. *Computer Speech & Language*, 40, 60–78.

[12] Abdul Sittar and Iqra Ameer. 2018. Multi-lingual author profiling using stylistic features. In *FIRE (Working Notes)*, 240–246.

[13] Abdul Sittar, Hafiz Rizwan Iqbal, and Rao Muhammad Adeel Nawab. 2016. Author diarization using cluster-distance approach. In *CLEF (Working Notes)*. Citeseer, 1000–1007.

[14] Abdul Sittar and Dunja Mladenic. 2021. How are the economic conditions and political alignment of a newspaper reflected in the events they report on? In *Central European Conference on Information and Intelligent Systems*. Faculty of Organization and Informatics Varazdin, 201–208.

[15] Abdul Sittar, Dunja Mladenić, and Marko Grobelnik. 2022. Analysis of information cascading and propagation barriers across distinctive news events. *Journal of Intelligent Information Systems*, 58, 1, 119–152.

[16] Jie Xu, Huayi Tang, Yazhou Ren, Liang Peng, Xiaofeng Zhu, and Lifang He. 2022. Multi-level feature learning for contrastive multi-view clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16051–16060.

Automatically Generating Text from Film Material – A Comparison of Three Models

Sebastian Korenič Tratnik
Jožef Stefan International Postgraduate School
Faculty of Computer and Information Science
Večna pot 113
Ljubljana, Slovenia

Erik Novak
Jožef Stefan Institute
Jožef Stefan International Postgraduate School
Jamova cesta 39
Ljubljana, Slovenia

ABSTRACT

The paper focuses on audio analysis and text generation using film material as an example. The proposed approach is done by using three different models (Wav2Vec2, HuBERT, S2T) to process the sound from different audio-visual units. A comparative analysis shows the strengths of different models and factors of different materials that determine the quality of text generation for functional film annotation applications.

KEYWORDS

Text generation, automated transcription, cinema, film, video

1 INTRODUCTION

Applications like automatic text captions for video materials have become more and popular and extensively used by users on different media, spanning from the computer, television, smartphones and other technologies that enable audio-visual consumption. However, even though these applications have to an extent already become a staple in our everyday lives, their performance often varies and still has not reached optimal functionality. There are many challenges when we work with text generation out of audio-visual materials. These span from the structure and quality, the type or category of sound, the age of the recordings and the models on which such translation is based on. The main goal of this paper is to provide a practical demonstration of a few basic models for automatic annotation. The goal is to take into account the currently most common procedures of such an endeavour and figure out how to minimize the loss function of the models to allow an optimal generation of text out of film or video more sufficiently.

The rest of this paper is organized in the following way. Section 2 provides a description of the problem in the context of contemporary consumption of audio-visual materials via most popular information and communication technologies. Section 3 delineates the methodology used and describes the approach used to tackle the problem in a concrete demonstration. Section 4 presents the models being used and describes our implementation of them, specifying the dynamics of the obtained results. A conclusion is reached in section 5, where the paper offers a discussion on the outcome and possible directions for future work.

2 PROBLEM DESCRIPTION

In recent years, audio-visual data has become as influent if not more influent as traditional text-based information. With this, the task of extracting information from the former and transforming it into the latter is becoming useful for different purposes [1, 2]. One example is that text annotations enable better comprehension in cases of bad sound quality or even allow the material to be understood in situations where sound consumption is impossible. Another one is a possible speed up of the video that the annotations provide due to their ability to keep the content integral in a clear graphic form. The consumption process can be made more time efficient with textual information compensating for the distortions of audio-visual quality that can be brought about with the manipulations of playing options. Furthermore, in a general sense, combining audio-visual material with text can solve many problems on different levels of film or video production. This can span from the preparing phases of pre-production such as writing the script, to the post-production phases where one needs good orientation over a vast quantity of material. Proper text generation can facilitate easier orientation in such work and allows for more efficient organization of the media materials.

In this paper, we will focus on those components that contribute to the quality of proper automated text generation as a prerequisite of such developmental strategies. The main contributions of this paper are: (1) an analysis of the factors that influence automatic transcription of film or video material (2) implementation and comparison of a few different models for sound annotation (3) reflection on how this process can be used for more complex tasks

3 METHODOLOGY

The problem we are solving is to take a piece of audio-visual material, convert it into a code that a model for automatic text generation can take as input and then generate output of text that matches the sound recording of the input in an optimal way. An optimal result should provide a close correspondence of the utterances in the film material and eventually identify different types and categories of sound such as dialogue, noise, music etc. We will do an analysis of the factors that influence the quality of automatically generated transcriptions in the following steps: 1) a comparison of different models for generating text from audio files, 2) an analysis of how the quality of transcriptions differs in relation to noise in the background (silence, music, dialogues), 3) an evaluation of how the clarity of speech influences the quality of transcriptions, and 4) an assessment to what extent it is more difficult to generate quality transcriptions from older audio-inscriptions (films).

Reflecting on the results of our procedure, we will think about how to improve the quality in cases when quality of transcriptions is bad. Aside from quality we will measure the time demands of models, that is how much time do the models need to generate transcriptions from the audio writing.

The following model were used:

1) Wav2Vec2 [4] is a framework for self-supervised representation learning from raw audio that was made open-source by Facebook. It is the first Automatic Speech recognition model included in Transformers as one of the central parts of Natural Language Processing. Figure 1 shows the model's architecture.

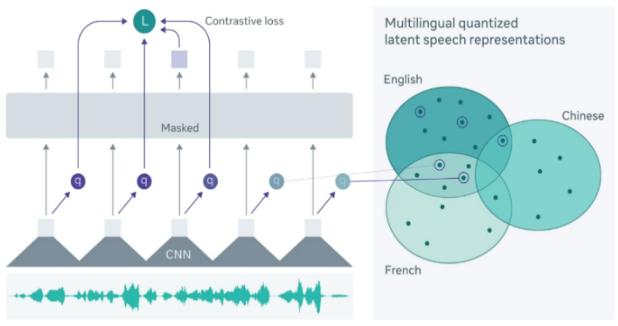


Figure 1. Wav2Vec2 learns speech units from multiple languages using cross-lingual training [4].

The model starts by processing the raw waveform with a multilayer convolutional neural network. This yields latent audio representations of 25ms that are fed into a quantizer and a transformer. From an inventory of learned units, the quantizer chooses appropriate ones, while half of the representations are masked before being used. The transformer then adds information from the whole of the audio sequence and with the output leads to solving the contrasting task with the model identifying the correct quantized speech units for the masked positions.

2) HuBERT [3] (Hidden-Unit BERT) is an approach for self-supervised speech representation that uses masking in a similar way and in addition adds an offline clustering step that provides aligned target labels for a prediction loss. This prediction loss is applied over the masked regions, which leads the model to learn a combined language and acoustic model over the continuous inputs. By focusing on the consistency of the unsupervised clustering step rather than the intrinsic quality of the assigned cluster labels, HuBERT can either match or improve the Wav2Vec2 model. Figure 2 shows the model's architecture.

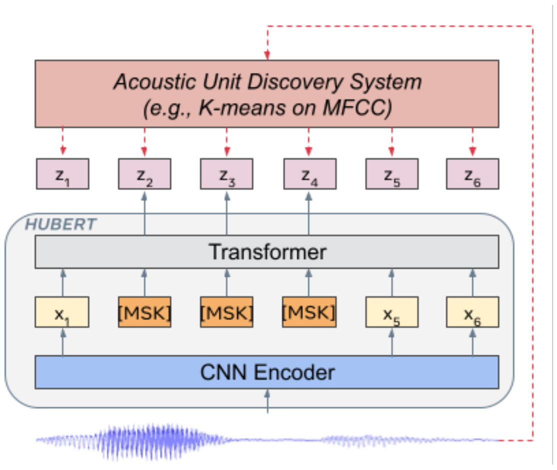


Figure 2. HuBERT predicts hidden clusters assignments using masked frames (y2, y3, y4 in the figure) generated by one or more iterations of k-means clustering [7].

3) S2T [5] (Speech2Text) is a transformer-based encoder-decoder (seq2seq) model that uses a convolutional downsampler to dramatically reduce the length of audio inputs over one half before they are fed into the encoder. It generates the transcripts autoregressively and is trained with standard autoregressive cross-entropy loss.

4 EXPERIMENT SETTING

4.1 Evaluation metric

We have used WER (Word error rate) as the metric of the performance of the models which computes the error rate on the comparison of substitutions, deletions, insertions and correct words. Original text was used for each of the model and each film example, removing the punctuation.

$$\mathbf{WER} = \frac{\mathbf{S + D + I}}{\mathbf{N}}$$

where...
S = number of substitutions
D = number of deletions
I = number of insertions
N = number of words in the reference

4.2 Data set

The dataset was formed with clips of different films. The films used were classics of world cinema (*The Godfather, 2001: A Space Odyssey, Star Wars, Frankenstein, Fight Club, Paris, Texas, Scent of A Woman, Tomorrow and Tomorrow and Tomorrow*). 14 clips of sizes spanning from 5 to 30 seconds were used with the lengthier ones incorporating different sound contents (like speech, shouting, whispering etc.). The first step was to prepare the audio in such a format that the models will be able to read it, so the clips were changed from mp4 to wav. An online converter, **cloudconvert** [https://cloudconvert.com], was used as the clips were fairly short and the results could be directly added to the Kaggle dataset from the browser itself.

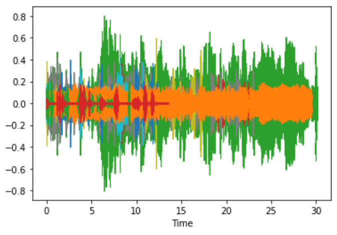


Figure 3: A superposition of waveform graphs of all the examples.

4.3 Implementation details

Programming was done on Kaggle, where code was written in Python and after the experiments were set up, and the GPU was activated for faster computation. The general process using each of the models is the following. First, an encoder takes raw data and puts it in the model. In our demonstration, tokenizers were used at the start, but as S2T tokenizers was not equipped to get the audio, it had to be changed to a processor. To retain consistency, the same step was applied to the other two models as well. Once data gets in the model, the model predicts particular syllables for each sound with certain probabilities and then in an additional step selects those with the highest probability based in the context of the semantic whole of the sentence. In the final step, the decoder (again the tokenizers / the processors) takes the output of the model and transforms it into text.

5 EXPERIMENT RESULTS

The ground rules for our project were that each model had a particular function that took sound as input and produced text as output with each audio having the text extracted separately. Subsequently different models were compared according to the accuracy of the results according to different criteria and a variety of scenarios (noise, music, number of characters, tempo of speech etc.). We will illustrate the obtained results via a concrete example. We will take a clip with relatively clear sound from the film *A Few Good Men* (1992), a digitized version of a well preserved celluloid film. The sound is clear and the dialogue takes places in a court practically in complete silence of the surroundings with the speech changing from normal tone to screaming. The clip is 22 seconds long and its waveform is shown in Figure 4. The original text is as following:

A: Did you order the Code Red?!
B: You don't have to answer that question!
C: I'll answer the question. You want answers?
A: I think I'm entitled!
C: You want answers!?
A: I want the truth!
C: You can't handle the truth! Son, we live in a world that has walls, and those walls have to be guarded by men with guns. Who's gonna do it? You? You, Lieutenant Weinberg?

The produced transcriptions are as follows:

Wav2Vec2:
YOU WAR THE CORA YOU DON'T HAVE TO ANSWER THE QU ESTION I'LL ANSWER THE QUESTION YOU WANT ANSWERS I THINK I'M ENTITLE YOU WANT ANT A AT THE TRUE YOU CA N'T HANDLE THE TRUTH SON WE LIVE IN A WORLD THAT H AS WALLS AND THOSE WALLS HAVE TO BE GUARDED BY ME N WITH GUNS WHO'S GON TO DO IT YOU YOU LIEUTENANT WINEBERG

HuBERT:
OMARTER TE CORET YOU DON'T HAVE TO ANSWER THAT Q UESTION I'LL ANSWER THE QUESTION YOU WANT ANSWERS I THINK I'M ENTITLED YOU WANT ANSWERRTHE TRUTH YO U CAN'T HANDLE THE TRUTH SON WE LIVE IN A WORLD TH AT HAS WALLS AND THOSE WALLS HAVE TO BE GUARDED B Y MEN WITH GUNS WHO'S GOING TO DO IT YOU YOU LIEUT ENANT WINBURG

S2T:
DEAR LORD THE CORRET YOU DON'T HAVE THE ANSWER T HAT QUESTION I'LL ANSWER THE QUESTION YOU WANT AN SWERS BUT THEY CAN'T ENTITLE YOU ONE AND THE TRUTH YOU CAN'T HANDLE THE TRUTH SOME WE LIVE IN A WORL D THAT HAS WALLS AND THOSE WALLS HAVE TO BE GUARD ED BY MEN WITH GUNS WHOSE TENANT DO IT YOU LIEUTE NANT WINEBURG THOSE HAVE TO BE GUARDED BY MEN WI TH GUNS WHOSE CANNON DO IT YOU YOU LIEUTENANT WI NEBURG YOU LIEUTENANT WINEBURG

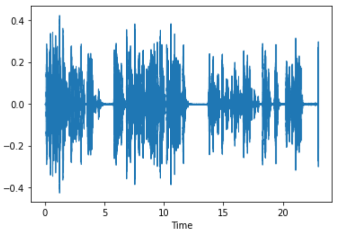


Figure 4. A scene from A Few Good Men (1992), a still and waveform graph from the used sequence.

The lower the WER number, the better the results. The models did not have a noticeable variation of speed, while the quality of their performance varied due to different factors. Hubert gave overall the best results from the point of view of readability. According to the rate of correspondence between input audio and output text, HuBERT comparably gave the better rate of the transcription in case of videos with poor audio quality from Wav2Vec2, i.e. that from older or damaged films, while Wav2Vec2 gave better performance in case of background music, but had the tendency of adding too much insertions. S2T had the tendency to produce mistakes, seen in peaking numbers over 1.0. The overall results are given in Table 1.

It is important to note that the average given does not reflect the better overall accuracy, but is the sum of different factors. So the models can be good at transcribing particular words, but can add or drop extra words in the process and therefore make the overall text less comprehensible. An important factor is the way the original text that is used for comparison is written – omitting punctuations and properly writing the words even if they are mispronounced will improve the results. Finally, it is crucial that all the texts are in caps lock, or the comparison won't work and will produce misleading results.

As the used example shows, it is mostly clarity of speech that will determine how the models perform. As the models were pre-trained and were not trained according to the specific data used, they were in general surprisingly efficient. The discrepancies in different treatments of the same audio are visible, but in general as long as the dialogue was clear, the results were comparable. Music seemed to cause bigger problems for the model than background noise, while additional speech in the background proved most problematic. Emotional influences on speech did not prove that problematic and even affective utterances were transcribed comparably with neutral speech if the sound data was of high quality.

Table 1. The WER scores for each model. The bold values represent the best performances on the given clip. The best performing model is HuBERT.

Clip number	Wav2Vec2	HuBERT	S2T
1	69%	**53%**	91%
2	100%	**0%**	100%
3	100%	**95%**	**95%**
4	**27%**	30%	36%
5	**17%**	**17%**	**17%**
6	39%	**18%**	43%
7	**28%**	**28%**	64%
8	70%	**46%**	55%
9	50%	**25%**	100%
10	57%	**37%**	73%
11	62%	**38%**	51%
12	100%	**95%**	100%
13	60%	**33%**	73%
14	9%	**4%**	9%
Average	56%	**37%**	65%

The WER usually shows the results in a metric between 0 and 1, however in case the annotation results were extremely unsuccessful, the higher extreme may surpass the limit. In our case, up to 1.6 was reached, however in the chart, it was limited down to 1.0 for purposes of clarity.

5 DISCUSSION AND FURTHER WORK

So as a general principle, when taking clips from films, the main factor that can potentially influence the quality of the generated text in a negative way is the background noise. As one can expect, the model will work best when nothing is in the background and worst when people are talking in the background. Ideally, to improve the quality one would train the models for the specific material, using a similar type of material and accordingly doing a pre-classification according to the main categories of sound analysis (ie. monologue, dialogue, background noise, music, echo, normal speech, loud speech, shouting, whispering etc.) - especially when using older or less preserved material, which drastically differs in sound data from newer or more preserved works.

In our research we expanded on and adapted existing work on automated text generation models, providing an analysis of the factors that determine the quality of such results from film material. As an example, we applied our approach on different film material, ranging in the quality and age of the clips and the structure of the sound data.

A useful strategy for the future from the perspective of film practice would be to find ways to link transcriptions with a script. A precondition of such an endeavour would be to implement an algorithm for recognizing the person speaking and identifying the source with descriptions ("person A is speaking, then person B, then person A has a long monologue, person C answers" etc.). Another important task would be identifying the sounds of different categories and providing fitting audio-signs (sound of squeaking steps, playing of music etc.). From these steps one could eventually at least to some extent automatically generate scripts for films or find ways to develop tools for easier text-based classification of audio-visual material.

CONCLUSIONS

In this paper we explored ways to generate text out of audio information presented in film and video material. We used three different models to evaluate various film units, Wav2Vec2, HuBERT, and S2T. We found that the model HuBERT achieved best results, while the remaining two methods performed similarly.

ACKNOWLEDGMENTS

The research described in this paper was supported by International Postgraduate School Inštitut Jožef Štefan, Ljubljana, Slovenia in the class Textual/ Multimedia mining and semantic technologies held by dr. Dunja Mladenič under the mentorship of Erik Novak. We also thank Besher Massri, Aljoša Rakita and Martin Abram for additional feedback.

REFERENCES

1 A. Ramani, A. Rao, V. Vidya and V. B. Prasad, "Automatic Subtitle Generation for Videos," *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2020, pp. 132-135, doi: 10.1109/ICACCS48705.2020.9074180.

2 Rustam Shadiev, Yueh-Min Huang, Facilitating cross-cultural understanding with learning activities supported by speech-to-text recognition and computer-aided translation, Computers & Education, Volume 98, 2016, Pages 130-141

3 Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, Abdelrahman Mohamed. HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. [arXiv:2106.07447v1 [cs.CL], Submitted on 14 Jun 2021].

4 Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, Michael Auli. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations.. [arXiv:2006.11477v3, Submitted on 20 Jun 2020 (v1), last revised 22 Oct 2020 (this version, v3)].

5 Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, Juan Pino. fairseq S2T: Fast Speech-to-Text Modeling with fairseq. [arXiv:2010.05171v1, Submitted on 11 Oct 2020] .

6 Wav2vec2.0: Learning the structure of speech from raw audio. [https://ai.facebook.com/blog/wav2vec-20-learning-the-structure-of-speech-from-raw-audio Submitted on 24 Sep 2020, Access. 9.1.2022

7 Hsu, Wei-Ning, et al. "Hubert: Self-supervised speech representation learning by masked prediction of hidden units." IEEE/ACM Transactions on Audio, Speech, and Language Processing 29 (2021): 3451-3460.

The Russian invasion of Ukraine through the lens of ex-Yugoslavian Twitter

Bojan Evkoski
bojan.evkoski@ijs.si
Jozef Stefan Institute, and
Jozef Stefan Postgraduate School
Ljubljana, Slovenia

Igor Mozetič
igor.mozetic@ijs.si
Jozef Stefan Institute
Ljubljana, Slovenia

Petra Kralj Novak
petra.kralj.novak@ijs.si
Central European University
Vienna, Austria, and
Jozef Stefan Institute
Ljubljana, Slovenia

Nikola Ljubešić
nikola.ljubesic@ijs.si
Jozef Stefan Institute, and
Faculty of Computer and Information Science,
University of Ljubljana
Ljubljana, Slovenia

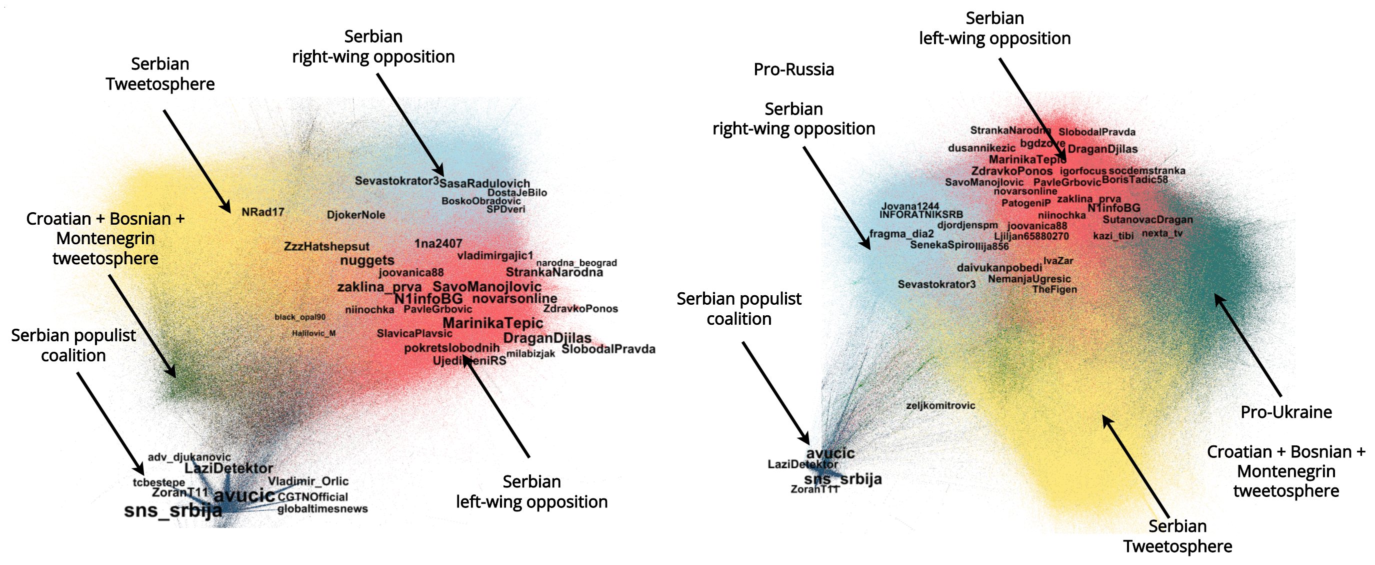

Serbian Tweetosphere
Serbian right-wing opposition
Croatian + Bosnian + Montenegrin tweetosphere
Serbian populist coalition
Serbian left-wing opposition
Pro-Russia
Pro-Ukraine
avucic
sns_srbija
LaziDetektor
N1infoBG
DraganDjilas
SavoManojlovic
MarinikaTepic

Figure 1: Pre-invasion (left) and invasion (right) ex-Yugoslavian retweet networks. Node colors represent communities. Labeled arrows point to the main communities, with labels inferred from the community users. The in-network labels represent the names of the most retweeted accounts.

ABSTRACT

The Russian invasion of Ukraine marks a dramatic change in international relations globally, as well as at specific, already unstable, regions. The geographical area of interest in this paper is a part of ex-Yugoslavia where the BCMS (Bosnian, Croatian, Montenegrin, Serbian) languages are spoken, official varieties of a pluricentric Serbo-Croatian macro-language [4]. We analyze 12 weeks of Twitter activities in this region, six weeks before the invasion, and six weeks after the start of the invasion. We form retweet networks and detect retweet communities which closely correspond to groups of like-minded Twitter users. The communities are distinctly divided across countries and political orientations. Some communities detected after the start of the Russian invasion also show clear pro-Ukrainian or pro-Russian stance. Such analyses of social media help in understanding the role and effect of this conflict at the regional level.

KEYWORDS

social network analysis, community detection, Twitter

1 INTRODUCTION

The Russian invasion of Ukraine brings about dramatic changes to the world. Analysing the structure and content of the communication on social media, such as Twitter, can give more insight into the causes, developments and consequences of this conflict. The geographical area of interest in our research is a part of ex-Yugoslavia where the BCMS (Bosnian, Croatian, Montenegrin, Serbian) languages are spoken, official varieties of the pluricentric Serbo-Croatian macro-language. This area is strongly politically divided by diverging influences of NATO (Croatia, Montenegro, North Macedonia, Bosniak and Croatian entity in Bosnia and

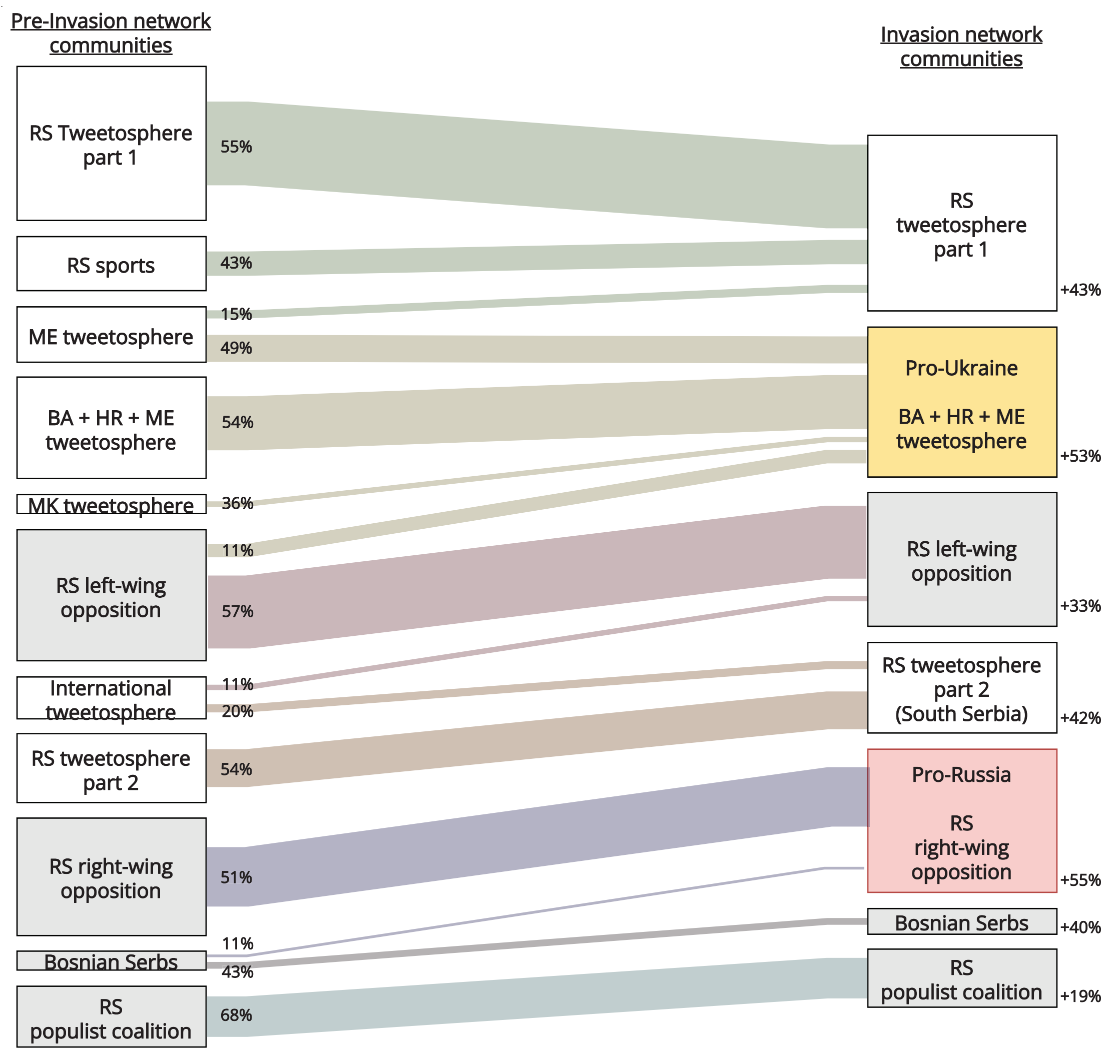


Figure 2: A Sankey diagram showing the transitions of users from the pre-invasion network communities (left) to the invasion network communities (right). Rectangle height is proportional to the community sizes. Percentages near the pre-invasion communities show the portion of users found in the corresponding invasion communities. Percentages on the right-hand side of the invasion communities show the portion of users not previously present in the large communities of the pre-invasion network. Gray rectangles depict the communities tightly related to politics, with the yellow and red denoting the detected pro-Ukraine and pro-Russia leaning communities, respectively.

Herzegovina) and Russia (Serbia, Serbian entity in Bosnia and Herzegovina). While Croatia is full EU member since 2013, Montenegro, North Macedonia and Serbia are EU candidate members, while Bosnia and Herzegovina is a potential candidate. Regarding military alliances, NATO members are Croatia (since 2007), Montenegro (since 2017) and North Macedonia (since 2020), while Serbia does not aspire to join NATO, primarily due to a complex Serbia-NATO relationship caused by the NATO intervention in Yugoslavia in 1999.

To shed light on the impact of the Russian invasion on this brittle and complex geographical and political area, we use social network analysis over available Twitter data, 6-weeks before and 6-weeks during the invasion. We discover a complex landscape of ideology-specific and country-specific communities (see Figure 1), and analyse the transition into evident pro-Ukraine and pro-Russia leanings. We also present a method to measure the similarity of the communities before and during the invasion by analyzing URL and hashtag usage. As the communities show very divergent properties, we echo concerns of the heavy polarization and possible destabilization of this area of the Balkans.

2 RESULTS

The data analysed in this study were collected with the TweetCat tool [3], focused on harvesting tweets of less frequent languages. TweetCat is continuously searching for new users tweeting in the language of interest by querying the Twitter Search API for the most frequent and unique words in that language. Every user identified to tweet in the language of interest is continuously collected from that point onward. This data collection procedure is run for the BCMS set of languages since 2017. During the 12 weeks of our focus, we collected 1.2M tweets and 3.8M retweets from 45,336 users. A rough estimate of the per-country production of tweets via URL usage from country-specific top-level domains (upper part of Table 1) shows for Twitter to be much more popular in Serbia and Montenegro than in Croatia or Bosnia and Herzegovina. This has to be taken into account while analysing the communities of the underlying tweetosphere.

We created **pre-invasion** and **invasion retweet networks** (users as nodes, retweets as edges) from the collected data. We applied community detection (Ensemble Louvain [1]) on the two

Country	Population	URLs
Serbia (RS)	**7.2M (47.3%)**	**106K (44.2%)**
Croatia (HR)	3.9M (25.6%)	19.6K (8.1%)
Bosnia and Herzegovina (BA)	3.5M (23.0%)	14.9K (6.2%)
Montenegro (ME)	620K (4.1%)	24.7K (10.2%)
Total	15.2M	242K

Pre-invasion communities	Users	Tweets	Retweets	Intra-com. RTs
RS tweetosphere part 1	**13K (29.0%)**	**125K (24.9%)**	300K (18.9%)	80.3%
RS tweetosphere part 2	2.5K (5.6%)	35.8K (7.1%)	63.2K (4.0%)	62.3%
RS sports	1.6K (3.6%)	12.6K (2.5%)	25.6K (1.6%)	53.8%
ME tweetosphere	1.7K (3.8%)	22.7K (4.5%)	44.6K (2.8%)	74.5%
BA + HR + ME tweetosphere	5.6K (12.4%)	37.8K (7.5%)	59K (3.7%)	75.3%
Macedonian tweetosphere	200 (0.4%)	721 (0.1%)	771 (0.1%)	77.7%
International tweetosphere	934 (2.0%)	8.5K (1.7%)	11.5K (0.7%)	62.3%
RS populist coalition	2.0K (4.8%)	52.4K (10.4%)	396K (24.9%)	**98.7%**
RS left-wing opposition	9.3K (20.6%)	105K (20.9%)	**408K (25.5%)**	80.5%
RS right-wing opposition	7.6K(16.8%)	87.8K (17.4%)	247K (15.5%)	72.1%
Bosnian Serbs	139 (0.3%)	2.2K (0.4%)	3.8K (0.2%)	83.1%
Total	45.3K	502.9K	1590K	

Invasion communities	Users	Tweets	Retweets	Intra-com. RTs
RS tweetosphere part 1	**16.9K (29.5%)**	160K (22.4%)	387K (16.8%)	71.1%
RS tweetosphere part 2	4.5K (7.7%)	57.3K (8.1%)	118K (5.1%)	58.1%
Pro-Ukraine BA + HR + ME tweetosphere	12.4K (21.7%)	76.1K (10.6%)	235K (10.2%)	64.7%
Pro-Russia RS right-wing opposition	11.1K (19.4%)	129K (17.9%)	508K (22.1%)	65.1%
RS populist coalition	1.8K (3.1%)	**208K (29.1%)**	450K (19.5%)	**95.6%**
RS left-wing opposition	9.8K (17.2%)	191K (26.7%)	**590K (25.6%)**	72.6%
Bosnian Serbs	356 (0.6%)	5.4K (0.7%)	7.1K (0.3%)	62.3%
Total	57.4K (+26.7%)	717K (+42.8%)	2302K (44.8%)	

Table 1: The first part shows general population of each BCMS country and their respective tweet URL shares (.rs, .hr, .ba and .me). The second part shows the pre-invasion network communities with the number of users, tweets, retweets and intra-community retweets. The third part shows the same statistics for the invasion network communities. Grey rows depict political communities, while yellow and red show the pro-Ukraine and pro-Russia communities, respectively.

networks and analysed the community properties and user transitions [2]. We identified and named the large communities (more than 100 users) by a careful analysis of their most influential users and hashtag/URL usage. Figure 2 depicts the user transitions between the two networks, while Table 1 shows general statistics of each community. We discovered the following peculiarities:

- The BCMS tweetosphere is dominated by Serbian (RS) users and content.
- The political communities are more active compared to the non-political ones.
- RS populist coalition community (led by the Serbian president Aleksandar Vučić) forms a very strong echo chamber, with less than 2% of all users, yet more than 25% of tweets and retweets and more than 95% of intra-community retweets.
- RS populist coalition and left-wing opposition remain neutral on the invasion topic.
- RS right-wing opposition and the Bosnian Serbs show a clear pro-Russia stance.
- Croatian, Bosnian and Montenegrin communities show a clear pro-Ukraine stance.

In order to compare the pre-invasion and invasion communities in terms of content and political leanings, our following goal was to compare the pool of hashtags used and URLs shared by the community users. Therefore, we developed a simple community similarity method. First, we preprocessed the URLs by manually filtering out the ones coming from social media sources like Twitter, Facebook, Youtube etc., as well as URL shorteners. With this, we created a subset in which more than 99% of the URLs were news media, making it ideal for media polarization analysis. Once we extracted the domain of the URLs, we then created sorted lists of the top 50 URL domains and top 50 hashtags for each community, sorted by the usage counts. Finally, in order to calculate the similarities between communities, we used the Rank-biased overlap (RBO) measure for indefinite rankings [5].

We found out that the matchings between the pre-invasion and invasion communities based on highest-user-overlap transitions are also visible through the URL and hashtag similarities (see Figure 3). In fact, for each pre-invasion community, its respective highest-user-overlap invasion community is also the highest RBO pair for both URLs and hashtags. In other words, there is a strong positive correlation between the user transition percentages (Figure 2) and the RBO scores. E.g., 68% of the users from the pre-invasion "RS populist coalition" community transition in the "RS populist coalition" community in the invasion network. Meanwhile, The URL RBO of this pair is 0.64, while the hashtag RBO is 0.43, both as the highest combination for the pre-invasion "RS populist coalition" community, clearly matching it with its invasion transition-based counterpart. This shows that our simple similarity method based on URLs and hashtags can even help in better matching communities in the task of community evolution [6].

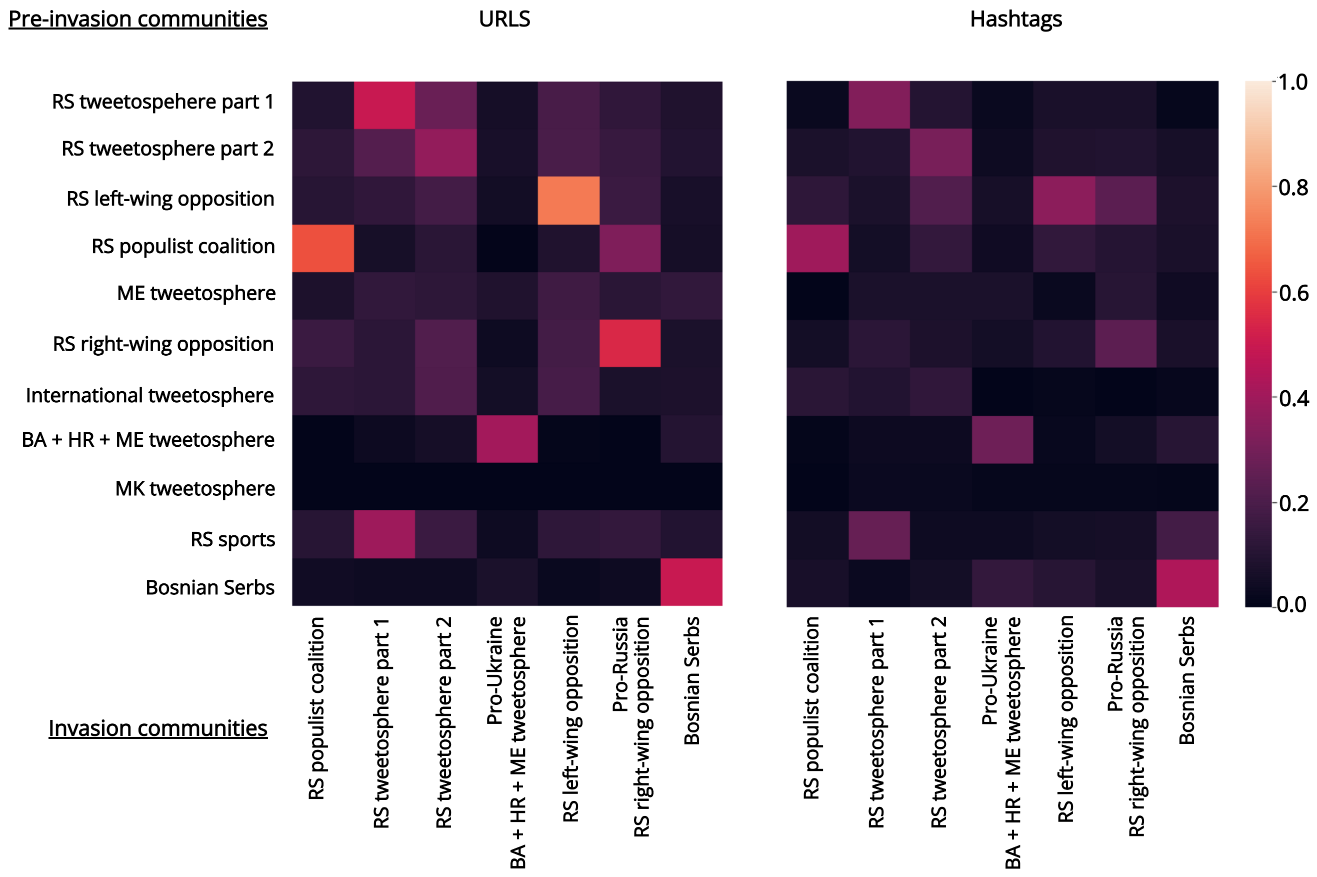


Figure 3: Domain and hashtag community similarities. A heatmap showing the similarities between the pre-invasion and invasion network communities based on the top 50 URLs (left) and hashtags (right). Similarities are calculated using the Rank-biased overlap (RBO) measure for indefinite rankings [5].

3 CONCLUSION

In this work, we investigated the Russian invasion of Ukraine through the lens of Twitter in the ex-Yugoslavian region where Bosnian, Croatian, Montenegrin and Serbian are spoken. We analyzed 12 weeks of Twitter activities in this region, six weeks before the invasion, and six weeks after the start of the invasion. For each period, we created retweet networks and detected retweet communities. We followed the transition of users from the pre-invasion to the invasion period and analyzed these groups of like-minded Twitter users, discovering that they are distinctly divided across countries and political orientations. For the invasion network, we were also able to detect communities which show clear pro-Ukrainian and pro-Russian stance.

Another contribution was a simple method for comparing retweet network communities based on the content of the tweets. The method showed a strong correlation with the most prominent user transitions we formerly discovered.

A continuation of this work is to expand it to a multidisciplinary research, with the aim to meticulously analyze the polarized content between the communities in collaboration with domain experts who are knowledgeable in ex-Yugoslavian politics. Beyond obtaining interesting insights, we also aim to explore two frequent issues in using social media for societal analyses: (1) uptake bias of specific social networks across countries and communities, and (2) entanglement of the main event with other large-scale events.

ACKNOWLEDGMENTS

The authors acknowledge financial support of the Slovenian Research Agency (research core funding no. P2-103 and no. P6-0411).

REFERENCES

[1] B. Evkoski, I. Mozetič, P. Kralj Novak. Community evolution with Ensemble Louvain. In *10th Intl. Conf. on Complex Networks and their Applications*, Book of abstracts, pp. 58–60, Madrid, Spain, 2021.
[2] B. Evkoski, I. Mozetič, N. Ljubešić, and P. Kralj Novak. Community evolution in retweet networks. *PLoS ONE*, 16(9):e0256175, 2021. Non-anonymized version available at https://arxiv.org/abs/2105.06214.
[3] N. Ljubešić, D. Fišer, and T. Erjavec. TweetCaT: a tool for building Twitter corpora of smaller languages. In *Proc. 9th Intl. Conf. on Language Resources and Evaluation*, pp. 2279–2283, ELRA, Reykjavik, Iceland, 2014.
[4] N. Ljubešić, M. Miličević Petrović, and T. Samardžić. Borders and boundaries in Bosnian, Croatian, Montenegrin and Serbian: Twitter data to the rescue. Journal of Linguistic Geography 6:2, DOI 10.1017/jlg.2018.9, pp 100-124, Cambridge University Press, 2018
[5] W. Webber, A. Moffat, Alistair, and J. Zobel. A similarity measure for indefinite rankings. ACM Trans. Information Systems 28(4):20, 2010.
[6] G. Rossetti and R. Caxabet. Community discovery in dynamic networks: a survey. ACM computing surveys (CSUR) 51.2 (2018): 1-37.

Visualization of consensus mechanisms in PoS based blockchain protocols

Daniil Baldouski
University of Primorska
Koper, Slovenia
d.baldovskiy@mail.ru

Aleksandar Tošić
University of Primorska
Koper, Slovenia
Innorenew CoE
Izola, Slovenia
aleksandar.tosic@upr.si

ABSTRACT

In the past decade, decentralized systems have been increasingly gaining more attention. Much of the attention arguably comes from both financial, and sociological acceptance, and adoption of blockchain technology. One of the frontiers has been the design of new consensus protocols, topology optimisation in these peer to peer(P2P) networks, and gossip protocol design. Analogue to agent based systems, transitioning from the design to implementation is a difficult task. This is due to the inherent nature of such systems, where nodes or actors within the system only have a local view of the system with very little guarantees on availability of data. Additionally, such systems often offer no guarantees of a system wide time synchronisation. This research offers insight into the importance of visualisation techniques in the implementation phase of vote based consensus algorithms, and P2P overlay network topology. We present our custom visualisations, and note their usefulness in debugging, and identifying potential issues in decentralized networks. Our use case is an implementation of a blockchain protocol.

KEYWORDS

Grafana, visualisation, consensus mechanism, blockchain protocols, P2P, overlay network

1 INTRODUCTION

Distributed systems are notoriously difficult to inspect and their problems difficult to identify. The difficulty stems from the fact that predominant issues can be stochastic and difficult to reproduce, and from the inability to easily observe, compare, and test multiple programs running on separate machines at the same time. Another important aspect in distributed systems is that they inherently make heavy use of the network. The use of various network protocols imposes additional complexity, which increases the search space in identifying bugs. In recent years, distributed systems have been gaining more attention both in academia and private sector. This increasing interest can be largely attributed to the rapid development of distributed ledger technology, and blockchain. In recent years, many new consensus mechanisms, blockchain protocols, network protocols, improvement in gossip protocols have been proposed. Many of them are transitioning from a theoretical framework to a practical implementation. However, public distributed ledgers (or distributed ledger technology or DLT) and blockchains secure their consensus mechanisms and provide spam resistance through the use of tokens representing value. The use of digital value within the protocol enables the protocol to enforce a level of security through economic incentives, and game theoretical aspects that make most attack vectors economically infeasible or impractical for the attacker. A good example of this is the Proof of Stake (PoS) consensus mechanism, where nodes in the decentralized protocol secure the consensus mechanism by requiring nodes to stake and lock up a considerable amount of value, which can be deducted (usually refereed to as slashing) by the protocol in case the node misbehaves. The economic aspect of public blockchains poses a very high security risk. With such strong economic incentives to identify and exploit potential bugs, and system faults, it is of upmost importance for the developers to thoroughly test and examine potential problems. However, the aforementioned difficulties in debugging distributed and decentralized protocols require developers to be equipped with tools that supports their efforts.

In this study, we review the state of the art approaches in testing and debugging voting based consensus mechanisms and decentralized networks. We develop a visualisation specifically designed for researchers and developers to test such networks and compare real-time observed data with the expected. We conclude that visualisation techniques can be complementary to traditional log based debugging, and testing techniques. Moreover, we provide our tools as open source software as plugins for popular visualisation platform Grafana. Both tools make no assumptions on the data storage implementation. The plugins can be configured via Grafana plugin configuration interface to fit the specifics of the protocol implementation. We validate our tools by applying them to a custom developed blockchain, and then explain how successful they turned out to be in identifying anomalies and bugs in the protocols.

2 THE ROLE OF VISUALIZATIONS IN DEBUGGING COMPLEX DISTRIBUTED SYSTEMS

Distributed and decentralized systems are difficult to debug as developers are working on the third layer. Which includes L1 (code level bugs), issues with concurrency on L2 (individual runtime), and finally the third dimension for potential bugs arising from the message exchange between nodes. In general, it is often hard to capture the state in a distributed system as debuggers cannot be attached to all nodes' run-times. Additionally, it is often difficult to reproduce errors when they are inherently stochastic. We consider several methods, such as *Logging, Remote debugging, Simulations* and *Visualisations.*

- Logging is the most common debugging method for all three layers. However, in distributed systems it is important to aggregate logs, and analyze them as a time series.

Additionally, aggregating distributed logs assumes the system has some method of clock synchronization protocol. Log collection has been proven to be effective in detecting performance issues for systems such as *Hadoop* [12] and *Darkstar* [13]. The aggregation can be done with specific tools for log collection such as *InfluxDB* [8], *Logstash* [10], etc. Aggregated logs then can be viewed in a form of a dashboard using tools like Grafana (see Figure 1).

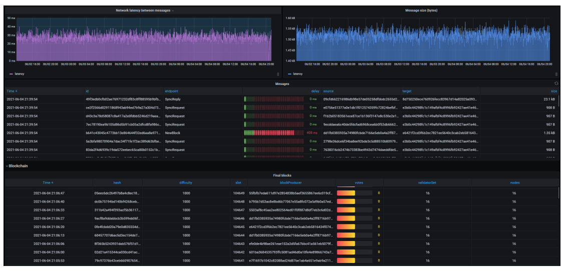


Figure 1: Part of the Grafana dashboard used by developers to gain insight into a running PoS based blockchain network.

- Remote debugging is a technique where a locally running debugger is connected to a remote node in the distributed system. This allows developers to use the same features as if they were debugging locally. However, it is difficult to determine which remote node should be debugged. Additionally, in case of Byzantine behaviour due to network faults connecting the debugger could fail.
- Distributed deterministic simulation and replay is a technique that attempts to address the issues of reproducibility in distributed systems. Tools like *Friday* [5] and *liblog* [6] can be used to record the specific state of the network to use and analyze it later. The technique suggests implementing an additional layer that abstracts the underlying hardware and the network interfaces to allow for an exact replay of all the state changes and messages exchanged between nodes. Tools such as FoundationDB or even custom systems are built on containerisation software.
- Visualisation and time series analysis attempts at capturing the state of the system, and all the nodes by visualising the collected logs. Tools like Prometheus [11] and Grafana [2] are used extensively. Tools like *Theia* [4] and *Artemis* [3] are designed for monitoring and analyzing performance problems in distributed systems and support built-in visualization tools for data exploration. However, such tools provide logs aggregated based summaries of the distributed systems and are not capable of observing underlying low-level network properties, e.g. monitoring network communication, especially in real-time while the system is running. *ShiViz* [1] on the other hand displays distributed system executions as an interactive timespace diagram. With this tool all the necessary events and interactions can be viewed in an orderly manner and inspected in detail. ShiViz visualization is based on logical ordering, meaning that unlike our tools, it is not capable of running in real-time, together with the considered distributed network. ShiViz also works with aggregated logs about various types of events of the distributed system and unlike our tools does not support direct database connections. ShiViz is generalized and works with all kind of distributed systems, while our tools are created specifically for monitoring PoS voting based consensus mechanisms and underlying network topology of the distributed system.

3 RESEARCH OBJECTIVES

The main goal of this research is to build visualisation tools that offer more insight into a running distributed system using the time series log collection data. The targeted system is a custom proof of stake based blockchain. Such tools should visualize if nodes contributing to the consensus learned about their correct roles, and if they perform their roles accordingly. In the consensus algorithms this is done by sending messages, so the tools should visualise messages exchanged between nodes.

In the structured P2P networks information spreads using gossip protocols and network topology changes every time slot. Our tools should visualize such changes in the network topology by drawing nodes and their cluster representatives, while at the same time indicating the consensus roles for each node.

In our implementation time series data comes from InfluxDB, but we want our tools to have no assumption on the data storage implementation and there are other popular databases, such as kdb+ and Prometheus, that work well with time series data. Because of that we choose Grafana as a platform for visualizations, which supports all of the aforementioned databases and many more at the time of writing.

In this work we implement two Grafana plugins built to visualize PoS based blockchains, and decentralized network topology. Our tools are designed with generality in mind, and are hence applicable to other PoS voting based blockchains and other distributed ledger implementations. We evaluate our tools by applying it to the custom developed blockchain and note their usefulness in debugging and identifying potential issues in decentralized networks.

4 GRAFANA PLUGINS FOR VISUALISING VOTE BASED CONSENSUS MECHANISMS AND P2P OVERLAY NETWORKS

We have developed two plugins that extend the functionality of Grafana. Figure 2 outlines the architecture used in production. A server running a database instance (preferably time series i.e. InfluxDB), and the Grafana platform. Depending on the underlying blockchain implementation, nodes can insert their telemetry directly to the database, or if possible have an archive node gather telemetry from nodes, and report them. In this example, a cluster was used to run multiple nodes. A coordinating node is responsible for maintaining an overlay network and serving the nodes within the overlay with a DHCP, DNS, and routing. Nodes are packed within docker containers and submitted to the coordinator, which uses built in load balancing and distributes them to other cluster nodes.

The telemetry inserted is timestamped to create a time series stream of data that is consumed by Grafana. Figure 1 shows a small part of the dashboard created within Grafana using the built-in plugins for typical visualisations. These visualisations are time series data of a running blockchain showing telemetry reported by the nodes. However, rendering telemetry from hundreds of nodes as factors is hardly informative.

Both plugins were developed as React components, using a well-known D3.js JavaScript library for animations and life-cycle of the plugins is managed by Grafana

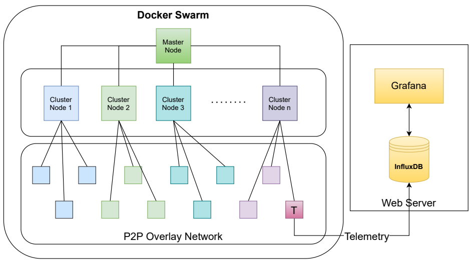


Figure 2: System architecture.

4.1 Network Plugin

P2P networks propagate information using gossip protocols. There are many variations of the general and implementation specifics but in general the family of protocols aims at gossiping the fact that new information is available in the network. Should a node hear about the gossip, and require the information it will contact neighbouring nodes asking for the data. In general, gossip protocols make no assumptions about the topology of the overlay network. However, with structured networks, the information exchange can be made much more efficient. The observed blockchain implementation utilized a semi structured network topology for propagating consensus based information. This is made possible by using a seed string shared between nodes that is used for pseudo-random role election every block. Using the seed, nodes self-elect into roles without the need to communicate. However, when performing roles, committee members must attest to the candidate block produced by the block producer. The seeded random is therefore also used to cluster the network using a k-means algorithm. The clustering is again performed by each node locally. The shared seed guarantees that nodes will produce the same topology, which is then used to efficiently propagate attestations to the block producer.

The network topology hence changes every slot. The plugin aims to visualize the changes in the network topology by drawing nodes, and their cluster representatives. Additionally, the consensus roles for each node are indicated with the vertex color. Figure 3 shows the network plugin rendering a test network of 30 nodes in real-time. The node in the center coloured green is the elected block producer for the current slot, nodes surrounded by the red stroke are cluster representatives, the rest of the nodes are coloured based on their role in the current slot.

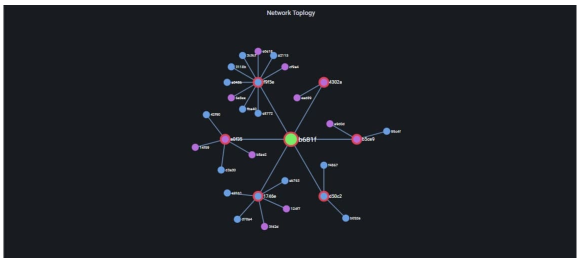
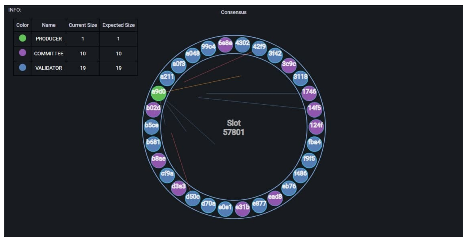


Figure 3: Network topology plugin visualising a test network of 30 nodes in real time.

4.2 Consensus Plugin

The aim of visualising the consensus mechanism is to quickly evaluate if nodes contributing to the consensus learned about their correct roles, and if they perform their roles accordingly. In order to have a scalable visualisation, nodes are placed around a circle, and scaled according to the size of the network. Roles are visualized with a color map. Each slot, nodes change their roles, and execute the protocol accordingly. To visualise the execution, the plugin visualises messages exchanged between nodes in a form of animated lines flying from an origin node to the destination node. The animations are time synchronous, and transfer times, and latencies are taken into account. Additionally, every message is logged with a type, indicating the sub protocol within which it was created. As an example, messages being sent from committee members to the block producer are attestations for the current block. The animated lines are coloured indicating the message type.

The thickness of the animated lines indicates the size of the payload transferred between nodes. Figure 4 shows the consensus plugin running live visualising a test network of 30 nodes. The green coloured node indicates the block producer role for the current slot, nodes coloured violet are part of the committee, and blue nodes are validators.


INFO:

Color	Name	Current Size	Expected Size
●	PRODUCER	1	1
●	COMMITTEE	10	10
●	VALIDATOR	19	19

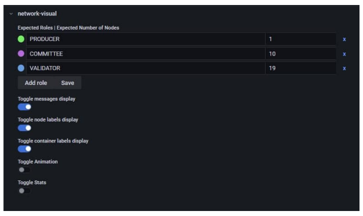
Figure 4: Consensus plugin (with legend) visualising a test network of 30 nodes in real time.

4.3 Generality

In order to use the above plugins, users have to provide certain data to the Grafana dashboard and this can be done through Grafana GUI. For the plugins to work all of the data should follow a specific naming policy. For example, for the Consensus plugin there is one necessary query to visualize data about the nodes of the network. It can be provided using SQL or Grafana GUI:

```
SELECT "slot", "node", "duty" FROM "<table-name>"
WHERE $timeFilter
```

Both plugins can be customized from the Grafana options menu. For example, users can add new roles, name and color them. Figure 5 shows the consensus plugin options menu, where users can additionally turn on or off display of messages, nodes or containers labels and so on. For both plugins, users have to manually provide the *slot time* of the network in the plugins options menus.


network-visual

Expected Roles | Expected Number of Nodes

●	PRODUCER	1	x
●	COMMITTEE	10	x
●	VALIDATOR	19	x

Toggle messages display

Toggle node labels display

Toggle container labels display

Toggle Animation

Toggle Stats

Figure 5: Consensus plugin options menu.

By using our tools we can visualize other protocols. For example with the consensus plugin we can visualize the famous Paxos algorithm, first introduced in [7] by Leslie Lamport. For that, we should provide the plugin with the *Nodes* and *Messages* queries. For the *Nodes* query, parameters *slot*, *node* and *duty* should be provided, which represent the slot number, node id and the role of the node respectively. From the point of nodes and slots, for this visualization Paxos works in the same way as the example of the PoS based consensus we mentioned before. For the *duty* parameter, nodes can have one of the three roles: *proposer*, *acceptor* or *learner*. That is why in the options menu of the plugin we should create 3 roles and name them according to the names from the data table.

We should specify *slot time* (in seconds) in the plugin options menu and at this point we can set the Grafana dashboard refresh time and see the results, since all the necessary conditions are fulfilled. But in order to gain more information from the plugin, we should add the *Messages* query. For the data we should have the following parameters: *id*, *source*, *target* and *endpoint*, which represent the message id, node id that sends the messages, node id that receives the message and the type of the message. For the additional information we can specify parameters *delay* (in seconds) and *size* of the message.

If we know the expected amount of nodes for some role, we can put it in the in plugin options menu to see this information in the plugin legend. In a similar way we should be able to visualize other consensus protocols, for example 2PC or Raft [9].

Source code for both plugins is open source, licensed under the MIT license and available on GitLab, where users can find the installation procedure of the plugins:

- Network plugin - https://gitlab.com/rentalker/topology-visualization-plugin,
- Consensus plugin - https://gitlab.com/rentalker/consensus-visualization-plugin.

5 CONCLUSION

We developed two Grafana plugins for visualising PoS based blockchains, and the underlying overlay network topology. The plugins were used to identify critical bugs, and faults in the protocol. With the help of visualisations, we were able to detect two problems when running test-nets.

- **Network congestion:** for every slot, validators must report their statistics to the block producer. Prompt delivery is desired but not critical. However, as the network grew in size, reporting statistics to a single node (block producer) became increasingly latent as all nodes attempted to propagate messages in tandem, and even more importantly, the network topology required a lot of routing for messages to arrive to the block producer. The network plugin helped us identify what the problem was by looking at the topology.
- **State synchronisation:** at random, nodes failed to perform their roles. This resulted in missing votes even on small test-nets, and sometimes a chain halt where no blocks were produced for the slot. We observed the likelihood of this happening grows in correlation with network size. However, it was infeasible to debug the state of all nodes in a large network. Visualising the state of nodes at a given slot we observed that states were not always synchronized and hence, some nodes did not learn about their consensus role.

We conclude that visualisation is an important tool in design and implementation of decentralized, and distributed systems. The methods serve a complementary role to existing debugging methods, and are very powerful at observing unexpected behaviour of the system as a whole. Visualisation techniques are specifically important in detecting stochastic faults that are non-trivial to reproduce. Our tools are open-source and available for researchers and engineers to use. They are suitable for testing any kind of voting-based consensus protocol with little effort.

For future work we would like to further develop our tools to accommodate other consensus protocols and help developers visualize and debug other types of issues related to distributed systems. Also, we would like to explore other types of visualizations and other existing tools that can help developers as well. Since Grafana is rapidly evolving, our developed plugins can be updated and new technologies can be integrated with our tools to improve their performance.

6 ACKNOWLEDGMENTS

The authors gratefully acknowledge the European Commission for funding the InnoRenew CoE project (H2020 Grant Agreement #739574) and the Republic of Slovenia (Investment funding of the Republic of Slovenia and the European Union of the European Regional Development Fund) as well as the Slovenian Research Agency (ARRS) for supporting the project number J2-2504 (C).

REFERENCES

[1] Beschastnikh, I., Wang, P., Brun, Y., and Ernst, M. D. Debugging distributed systems. *Commun. ACM 59*, 8 (jul 2016), 32–37.

[2] Chakraborty, M., and Kundan, A. P. Grafana. In *Monitoring Cloud-Native Applications*. Springer, 2021, pp. 187–240.

[3] Crețu-Ciocârlie, G. F., Budiu, M., and Goldszmidt, M. Hunting for problems with artemis. In *Proceedings of the First USENIX Conference on Analysis of System Logs* (USA, 2008), WASL'08, USENIX Association, p. 2.

[4] Garduno, E., Kavulya, S. P., Tan, J., Gandhi, R., and Narasimhan, P. Theia: Visual signatures for problem diagnosis in large hadoop clusters. In *Proceedings of the 26th International Conference on Large Installation System Administration: Strategies, Tools, and Techniques* (USA, 2012), lisa'12, USENIX Association, p. 33–42.

[5] Geels, D., Altekar, G., Maniatis, P., Roscoe, T., and Stoica, I. Friday: Global comprehension for distributed replay. vol. 7.

[6] Geels, D., Altekar, G., Shenker, S., and Stoica, I. Replay debugging for distributed applications. In *2006 USENIX Annual Technical Conference (USENIX ATC 06)* (Boston, MA, May 2006), USENIX Association.

[7] Lamport, L. The part-time parliament. *ACM Transactions on Computer Systems 16, 2 (May 1998), 133-169. Also appeared as SRC Research Report 49. This paper was first submitted in 1990, setting a personal record for publication delay that has since been broken by [60].* (May 1998). ACM SIGOPS Hall of Fame Award in 2012.

[8] Naqvi, S. N. Z., Yfantidou, S., and Zimányi, E. Time series databases and influxdb. *Studienarbeit, Université Libre de Bruxelles 12* (2017).

[9] Ongaro, D., and Ousterhout, J. In search of an understandable consensus algorithm. In *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference* (USA, 2014), USENIX ATC'14, USENIX Association, p. 305–320.

[10] Sanjappa, S., and Ahmed, M. Analysis of logs by using logstash. In *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications* (Singapore, 2017), S. C. Satapathy, V. Bhateja, S. K. Udgata, and P. K. Pattnaik, Eds., Springer Singapore, pp. 579–585.

[11] Turnbull, J. *Monitoring with Prometheus*. Turnbull Press, 2018.

[12] Xu, W., Huang, L., Fox, A., Patterson, D., and Jordan, M. Online system problem detection by mining patterns of console logs. In *2009 Ninth IEEE International Conference on Data Mining* (2009), pp. 588–597.

[13] Xu, W., Huang, L., Fox, A., Patterson, D., and Jordan, M. I. Detecting large-scale system problems by mining console logs. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles* (New York, NY, USA, 2009), SOSP '09, Association for Computing Machinery, p. 117–132.

Using Machine Learning for Anti Money Laundering

Gregor Kržmanc
gregor.krzmanc@ijs.si
Jožef Stefan Institute
Ljubljana, Slovenia

Filip Koprivec
filip.koprivec@ijs.si
Jožef Stefan Institute
IMFM
Ljubljana, Slovenia

Maja Škrjanc
maja.skrjanc@ijs.si
Jožef Stefan Institute
Ljubljana, Slovenia

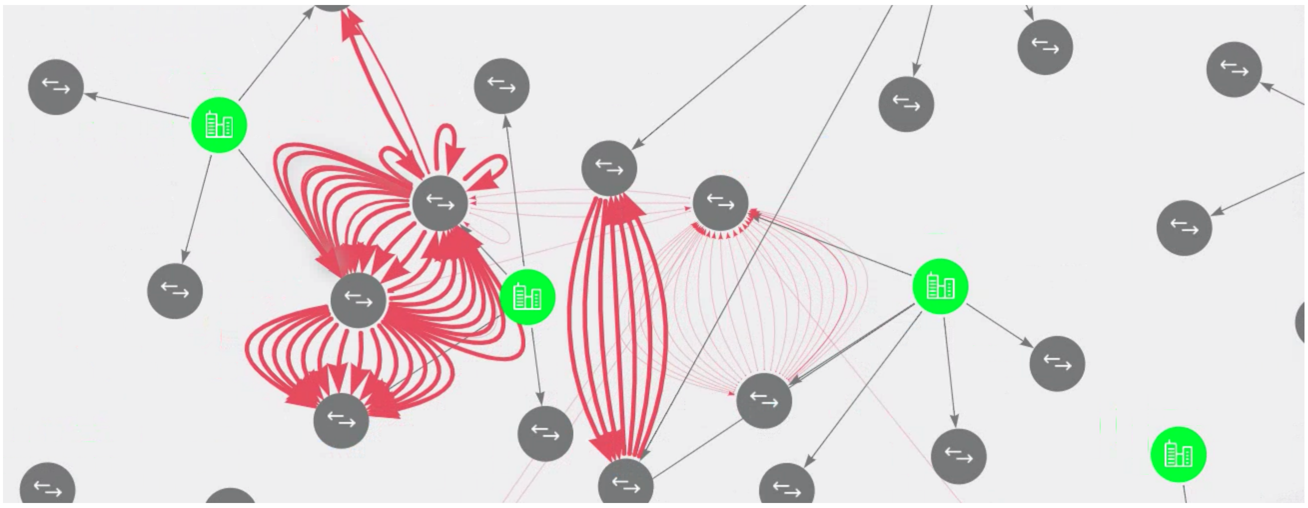


Figure 1: Example transaction network visualization

ABSTRACT

Here we present early results of a network component for anomaly detection in an attributed heterogeneous financial network. Utilizing both externally provided features and generated topological features, we train different models for a simple link prediction task. We then evaluate the models using initial dataset corruption. We show that gradient boosting and multi-layer perceptron generally have the best anomaly detection performance, despite graph neural network models initially showing better results in the link prediction task.

KEYWORDS

Anti Money Laundering (AML), machine learning, networks, link prediction

1 INTRODUCTION

Observing complex real-world graphs, be it a social, financial, biochemical, or physics-related network, is an interesting task. Given a time-evolving network and rich information about the nodes and edges, can we assume that there are some regular dynamics in the network?

Fraud and financial crime are important issues of our time. According to the United Nations Office on Drugs and Crime, an estimated 2-5 % of the world GDP is laundered each year. To keep pace with evolving trends, the European Union has decided to strengthen its anti money laundering and terrorist financing regulatory framework and expects the same from financial institutions and supervisory authorities.

Given a pseudonymized dataset of financial transactions, can we use machine learning to detect interesting, perhaps novel, patterns that should be inspected manually? In this paper, we try to answer this question.

2 RELATED WORK

Both supervised [7, 6, 12] and unsupervised or self-supervised [2, 14] learning approaches have been proposed to deal with the task of detecting money laundering. Due to the lack of labelled data and the closed nature of financial data and, therefore, the lack of standardised datasets, approach evaluation can be difficult. Despite that, cryptocurrency datasets such as [13] have been published, explored, and labelled to some extent.

Usually, synthetic oversampling or other strategies of sampling need to be employed in cases where labelled entities are used for evaluation [12, 13].

3 DATA

In this study, we use a snapshot of the transaction data processed through the international payment system *Target2-Slovenija* [11]. The dataset spans from November 2007 to December 2017, containing around 8 million financial transactions. No live data was used when performing this research - only archived datasets were used.

For some nodes, the data about the sending or receiving party is additionally linked to data from the Slovenian Business Register (ePRS) [1] and the Slovenian Transaction Account Registry (eRTR) [3] in order to provide additional context about each transaction.

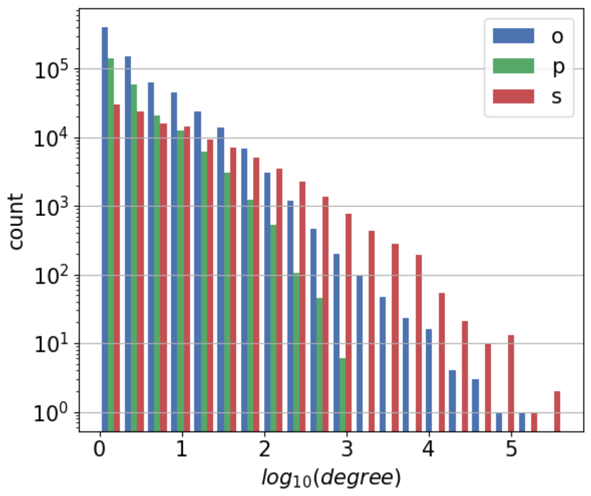


Figure 2: Degree distribution by node type.

Due to the sensitive nature of the data, all personal and confidential data about individuals and legal entities provided to JSI is pseudonymized.

4 DATA REPRESENTATION AS A HETEROGENEOUS GRAPH

There are large differences in the availability of data across different entities performing the transactions. In order to fully utilize all available features, we model the network as a heterogeneous temporal graph. Here, we treat the snapshot of the transaction graph from t_0 to t_1 $G = G(t_0, t_1)$ as a heterogeneous graph consisting of 3 discrete node types representing each entity's legal status. The types of accounts are those belonging to companies (node type s), natural persons (node type p), and all other accounts (node type o). Each transaction is represented as a directed edge from its source account to its destination account.

4.1 Network statistics

Due to different legislative bases for different types of entities, inherent differences regarding data availability are expected. Naturally, it is also expected that different categories usually act differently in a network - for example, companies usually transact more than individuals. While the degree distribution (Figure 2) closely resembles the power law, significant differences in distributions between different node types can be observed, which can be attributed to varying amounts of data available for our specific data source across account profiles.

It can be seen from Figure 2 that companies (node type s) perform most of the transactions.

4.2 Feature generation

Categorical features are one-hot encoded. Rare categories with $< 2\%$ incidence are marked as *other.* Additionally, node features encoding the role of a node in the network (Table 1) are generated. The node-level features for each node are computed on the whole network as well as for the subgraph induced by the node's own type.

feature	level
degree $\deg(A) = \|N(A)\|$	node-level
PageRank [9] $PR(A) = \frac{1-d}{N} + d \sum_{J \in N_{in}(A)} \frac{PR(J)}{\|N_{out}(J)\|}; d = 0.85$	node-level
Jaccard coefficient $J(A,B) = \frac{\|N(A) \cap N(B)\|}{\|N(A) \cup N(B)\|}$	edge-level
Adamic-Adar Index $A(x,y) = \sum_{u \in N(x) \cap N(y)} \frac{1}{\log \|N(u)\|}$	edge-level

Table 1: The structural features used for the link prediction task. $N(\cdot)$ represents the set of neighbours of the given node. N_{in} and N_{out} represent the sets of the nodes from which there is an edge to the given node (*in*), or to which there is an edge from the given node (*out*). $|\cdot|$ represents cardinality of the given set.

5 ANOMALY DETECTION PROBLEM DEFINITION

We corrupt the original graph by rewiring the total of $p = 1\%$ randomly picked edges of each edge type.

Let $f : V \times V \longrightarrow [0, 1]$ be a binary link prediction classifier that is trained to predict the probability that a directed edge between the two given nodes exists.

We define the anomaly score of edge $(i, j) \in E$ as

$$\phi(i,j) = 1 - f(i,j) \quad (1)$$

The intuition behind equation 1 is that links that are typical to the model would have a smaller anomaly score than links for which the model predicts they would not exist (and are, thus, anomalous).

6 RESULTS

We train several models for the downstream task of link prediction and then use the predictions for anomaly detection.

6.1 Experiment details

The traditional (non-GNN) machine learning approaches are trained to predict whether the given edge exists or not. For each edge, the feature vector fed into the model is constructed by concatenating source node features, destination node features, and edge features. For traditional models, a model for each edge type is constructed separately, while the graph neural network-based models are the same across all edge types.

The GNN (graph neural network) models are constructed of 2 layers of GraphSAGE aggregations [8, 5] using parametric ReLU activations and embedding dimensions of 128 for the first and 64 for the second layer. As messages are passed in the direction of edges, we construct another model to facilitate information diffusion both ways. We do this by adding edges of opposite directionality than existing edges and marking them as a separate edge types. We still, however, only train for the downstream link prediction objective only on the existing (non-transposed) edges. We mark this approach as GNN$^+$.

The traditional ML models used are gradient boosting (GradBoost), decision tree (DecTree), multi-layer perceptron (MLP) and logistic regression (LogReg). The hidden layer sizes of the MLP are 20 and 10, using ReLU activation in all layers except the last one, where softmax activation is used. Different combinations of reasonable hidden layer sizes were tested (32+16, 64+32, 256+128, 128+128, 20+10) and the best one was selected. The training of MLP models was performed with a batch size of 200.

6.2 Link prediction

Traditional ML models for link prediction map concatenated source and destination node features and edge features to the probability that a link between such nodes exists. The models are implemented using scikit-learn [10] and are trained and evaluated using 5-fold cross-validation.

As a preprocessing step, each feature is scaled individually using a standard scaler such that it has a mean of 0 and a standard deviation of 1 across the training set.

When training and evaluating each model, an approximately equal number of positive and negative links is given to the classifier. The provided edge features such as transaction amount are sampled randomly for negative edges.

Additionally, we train a 2-layer graph neural network (GNN) for link prediction. The GNN model is trained jointly for all edge types using weighted binary cross-entropy loss. The model has ReLU activations in all layers except the last one, where it has softmax activation. The hidden layer sizes are 64 and 32. The graph neural network is implemented using PyTorch Geometric [4].

We use a random link split for link prediction and not a temporal one, as our end goal is not to predict future links, but rather to learn what kinds of transactions are typical in the given network.

Table 2 shows the aggregated link prediction results. Bold results highlight the best performance across observed methods. The GNN does slightly improve link prediction performance in some cases. See Appendix A for more detailed non-GNN method results. The data here is computed across multiple year-long time windows.

edge	non-GNN	no str. f.	GNN	GNN$^+$
ss	0.92 ± 0.01	0.89 ± 0.01	0.92 ± 0.02	**0.94 ± 0.01**
oo	**0.80 ± 0.02**	0.57 ± 0.01	0.79 ± 0.02	0.53 ± 0.04
so	0.83 ± 0.01	0.75 ± 0.01	**0.88 ± 0.02**	0.74 ± 0.04
os	0.76 ± 0.01	0.64 ± 0.01	0.81 ± 0.01	**0.83 ± 0.02**
sp	**0.85 ± 0.02**	0.69 ± 0.03	0.78 ± 0.05	0.73 ± 0.02
ps	0.74 ± 0.02	0.67 ± 0.01	**0.87 ± 0.02**	0.75 ± 0.04
po	0.78 ± 0.02	0.66 ± 0.01	**0.84 ± 0.04**	0.54 ± 0.08
op	**0.89 ± 0.01**	0.53 ± 0.01	0.78 ± 0.05	0.50 ± 0.05
all	0.84 ± 0.01	0.72 ± 0.01	0.86 ± 0.02	**0.89 ± 0.01**

Table 2: Link prediction performance comparison measured in area under the receiver operating characteristic curve (AUC) (mean ± standard deviation). Edge types are marked with two letters, representing the source and destination node type in this order. Best non-GNN score, as well as best non-GNN score without using any structural features, are reported next to the GNN results.

6.3 Anomaly detection

For comparison between different methods, the 2% of edges with the highest anomaly scores are flagged as positive. Precision and recall are calculated by using the corrupted 1% of edges as true positives.

To summarize precision and recall in a single metric, F_1 score (2) is calculated and reported.

edge	non-GNN	no str. f.	GNN	GNN$^+$
ss	**0.19 ± 0.02**	0.16 ± 0.02	0.01 ± 0.00	0.01 ± 0.00
oo	**0.11 ± 0.02**	0.02 ± 0.01	0.05 ± 0.02	0.03 ± 0.02
so	**0.11 ± 0.02**	0.06 ± 0.01	0.01 ± 0.01	0.01 ± 0.01
os	**0.14 ± 0.02**	0.06 ± 0.01	0.01 ± 0.00	0.01 ± 0.01
sp	**0.08 ± 0.04**	0.02 ± 0.02	0.02 ± 0.01	0.02 ± 0.02
ps	**0.05 ± 0.02**	**0.05 ± 0.02**	0.01 ± 0.01	0.01 ± 0.01
po	**0.07 ± 0.04**	0.07 ± 0.05	0.02 ± 0.02	0.01 ± 0.02
op	**0.18 ± 0.04**	0.02 ± 0.01	0.02 ± 0.01	0.03 ± 0.02

Table 3: Anomaly detection performance comparison in F_1 score (mean ± standard deviation). Best non-GNN score, as well as best non-GNN score without using any structural features, are reported next to the GNN results. Bold results highlight the best performance across observed methods.

$$F_1^{-1} = \frac{\text{precision}^{-1} + \text{recall}^{-1}}{2} \quad (2)$$

A naive classifier that assigns the same positive score (recall 1) to each edge has F_1 score of ≈ 0.02. However, the underrepresented edge types typically have higher variance in F_1 score and performance insignificantly different from the naive baseline, as seen from Table 3. The same goes for the GNN-based models. See Appendix A for more detailed non-GNN model results.

7 DISCUSSION AND FUTURE WORK

We have constructed and evaluated a self-supervised approach to anomaly detection in financial networks. Due to the lack of labelled data, this is in most cases the most straightforward approach to tackle the problem with machine learning. There are significant differences in performance across different edge types. Using this approach yields almost comparable results with both raw features and structural features when evaluated on company-to-company transactions only. This may be explained by companies in our dataset having the most insightful features of all node types such as the broader sector and also more precise company industry type classification.

This paper has mainly focused on the use of unsupervised learning for anomaly detection. In the future, we plan to extend our work to supervised and semi-supervised learning approaches to try to utilize the few labelled data points. The following machine learning strategies (or a combination of them) could be tested:

- **Active learning.** Human-assisted active learning approach is a natural way to incorporate domain knowledge into the decision-making process.
- **Synthetic oversampling.** Due to a small number of the positive examples, we could sample new examples that are similar to them and assign them positive labels.
- **Model pretraining and few-shot learning.** Update model parameters with a self-supervised pretraining strategy first, and then optimize it further on the few labeled data points.

ACKNOWLEDGMENTS

The research leading to the results presented in this paper has received funding from the European Union's funded Project INFINITECH under grant agreement no. 856632.

The financial transaction data used in the presented research was collected and pseudonymized by the Bank of Slovenia.

The Bank of Slovenia collaborates with JSI and the Infinitech project in order to research possible efficient and compliant banking system supervision techniques.

We thank Klaudija Jurkošek Seitl for her input on the style of this paper.

REFERENCES

[1] 2022. AJPES - ePRS. (September 2022). https://www.ajpes.si/prs/.

[2] Claudio Alexandre and João Balsa. 2016. Client Profiling for an Anti-Money Laundering System. https://arxiv.org/abs/1510.00878.

[3] 2022. eRTR. (September 2022). https://www.ajpes.si/eRTR/JavniDel/Iskanje.aspx.

[4] Matthias Fey and Jan E. Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[5] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *arXiv:1706.02216 [cs, stat]*. arXiv: 1706.02216. Retrieved 06/18/2021 from http://arxiv.org/abs/1706.02216.

[6] Mikel Joaristi, Edoardo Serra, and Francesca Spezzano. 2019. Detecting suspicious entities in Offshore Leaks networks. *Social Network Analysis and Mining*, 9, 1, 1–15. Publisher: Springer Vienna. ISSN: 1327801906. DOI: 10.1007/s13278-019-0607-5. https://doi.org/10.1007/s13278-019-0607-5.

[7] Martin Jullum, Anders Løland, Ragnar Bang Huseby, Geir Ånonsen, and Johannes Lorentzen. 2020. Detecting money laundering transactions with machine learning. *Journal of Money Laundering Control*, 23, 1. DOI: 10.1108/JMLC-07-2019-0055. https://www.emerald.com/insight/1368-5201.htm.

[8] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*, (February 2017). arXiv: 1609.02907. Retrieved 06/18/2021 from http://arxiv.org/abs/1609.02907.

[9] Larry Page, Sergey Brin, R. Motwani, and T. Winograd. 1998. The PageRank Citation Ranking: Bringing Order to the Web. (1998).

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

[11] 2022. TARGET2 in TARGET2-Slovenija. si. (September 2022). Retrieved 09/21/2022 from https://www.bsi.si/placila-in-infrastruktura/placilni-sistemi/target2-in-target2-slovenija.

[12] Dominik Wagner. 2019. Latent representations of transaction network graphs in continuous vector spaces as features for money laundering detection. *Gesellschaft für Informatik*, 1–1.

[13] Mark Weber, Giacomo Domeniconi, Jie Chen, Daniel I Karl Weidele, Claudio Bellei, Tom Robinson, and Charles E Leiserson. 2019. Anti-Money Laundering in Bitcoin: Experimenting with Graph Convolutional Networks for Financial Forensics. Technical report.

[14] Jiaxuan You, Tianyu Du, Fan-yun Sun, and Jure Leskovec. 2021. Graph Learning in Financial Networks. (September 2021). https://snap.stanford.edu/graphlearning-workshop/slides/stanford_graph_learning_Finance.pdf.

A DETAILED RESULTS

A.1 Link prediction (AUC)

edge	DecTree	GradBoost	LogReg	MLP
ss	0.87 ± 0.01	0.90 ± 0.01	0.79 ± 0.01	**0.92 ± 0.01**
oo	0.80 ± 0.01	**0.80 ± 0.02**	0.51 ± 0.01	0.74 ± 0.01
so	0.82 ± 0.01	**0.83 ± 0.01**	0.65 ± 0.01	0.82 ± 0.01
os	0.75 ± 0.01	**0.76 ± 0.01**	0.58 ± 0.02	0.73 ± 0.01
sp	0.81 ± 0.02	**0.85 ± 0.02**	0.55 ± 0.02	0.83 ± 0.02
ps	0.70 ± 0.02	**0.74 ± 0.02**	0.54 ± 0.02	0.69 ± 0.01
po	0.72 ± 0.02	**0.78 ± 0.02**	0.54 ± 0.02	0.67 ± 0.01
op	0.85 ± 0.01	**0.89 ± 0.01**	0.51 ± 0.03	0.87 ± 0.01
all	0.81 ± 0.01	**0.84 ± 0.01**	0.66 ± 0.02	0.82 ± 0.01

A.2 Anomaly detection (F_1 score)

edge	DecTree	GradBoost	LogReg	MLP
ss	0.12 ± 0.01	0.13 ± 0.02	0.04 ± 0.01	**0.19 ± 0.02**
oo	0.07 ± 0.01	**0.11 ± 0.02**	0.01 ± 0.01	0.10 ± 0.02
so	0.08 ± 0.01	0.10 ± 0.02	0.04 ± 0.01	**0.11 ± 0.02**
os	0.06 ± 0.01	0.12 ± 0.02	0.04 ± 0.01	**0.14 ± 0.02**
sp	0.06 ± 0.01	0.07 ± 0.04	0.02 ± 0.02	**0.08 ± 0.04**
ps	0.04 ± 0.01	**0.05 ± 0.02**	0.01 ± 0.01	**0.05 ± 0.02**
po	0.04 ± 0.01	**0.07 ± 0.04**	0.02 ± 0.03	0.04 ± 0.03
op	0.09 ± 0.01	0.14 ± 0.04	0.01 ± 0.01	**0.18 ± 0.04**

Forecasting Sensor Values in Waste-To-Fuel Plants: a Case Study.

Bor Brecelj*
University of Ljubljana, Faculty of Mathematics and Physics
Ljubljana, Slovenia
bor.brecelj@gmail.com

Beno Šircelj*
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia
beno.sircelj@ijs.si

Jože M. Rožanec
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia
joze.rozanec@ijs.si

Blaž Fortuna
Qlector d.o.o.
Ljubljana, Slovenia
blaz.fortuna@qlector.com

Dunja Mladenić
Jožef Stefan Institute
Ljubljana, Slovenia
dunja.mladenic@ijs.si

ABSTRACT

In this research, we develop machine learning models to predict future sensor readings of a waste-to-fuel plant, which would enable proactive control of the plant's operations. We developed models that predict sensor readings for 30 and 60 minutes into the future. The models were trained using historical data, and predictions were made based on sensor readings taken at a specific time. We compare three types of models: (a) a näive prediction that considers only the last predicted value, (b) neural networks that make predictions based on past sensor data (we consider different time window sizes for making a prediction), and (c) a gradient boosted tree regressor created with a set of features that we developed. We developed and tested our models on a real-world use case at a waste-to-fuel plant in Canada. We found that approach (c) provided the best results, while approach (b) provided mixed results and was not able to outperform the näive consistently.

CCS CONCEPTS

• **Computing methodologies → Machine learning**; • **Applied computing**;

KEYWORDS

Smart Manufacturing, Machine Learning, Feature Engineering

ACM Reference Format:
Bor Brecelj, Beno Šircelj, Jože M. Rožanec, Blaž Fortuna, and Dunja Mladenić. 2022. Forecasting Sensor Values in Waste-To-Fuel Plants: a Case Study.. In *Ljubljana '22: Slovenian KDD Conference on Data Mining and Data Warehouses, October, 2022, Ljubljana, Slovenia.* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

There is a wide range of applications of ML (machine learning). One of them is the modeling and control of chemical processes, such as the production of biodiesel. Introducing machine learning to such processes can improve quality and yield and help engineers predict anomalies to control the factory better.

We modeled the JEMS waste-to-fuel plant, which produces high-quality diesel from organic waste. The plant has numerous sensors that measure temperature, and pressure, among other variables. It is operated by experts who must control the process. Since the chemical process is complex and, therefore, difficult to control, we built forecasting models that can predict future sensor readings based on historical data and the current state of the plant.

The model will be used to give plant operators additional information about the future state of the plant, which will allow them to make an informed decision about changing the plant's parameters and, therefore, adjust the process before it is too late.

*Both authors contributed equally to this research.

2 RELATED WORK

Organic wastes in energy conversion technologies are an active area of research aimed at reducing dependence on fossil fuels, optimizing production costs, improving waste management, and controlling emissions. Biochemical, physiochemical, and thermochemical processes produce different biofuels, such as bio-methanation, bio-hydrogen, biodiesel, ethanol, syngas, and coal-like fuels, which are studied by Stephen et al. [8]. Work is also being done on optimization, such as catalyst selection, reactor design, pyrolysis temperature, and other important factors [5].

Many ML methods have been developed to address waste management and proper processing for biofuel production, focusing on energy demand and supply prediction [3]. Aghbashlo et al. [2] provided a systematic review of various applications of ML technology with a focus on ANN (Artificial Neural Network) in biodiesel research. They provided an overview of the use of ML in modeling, optimization, monitoring, and process control. Models that predict the conditions of the biofuel production process that have the highest yield were created by Kusumo et al. [6] and Abdelbasset et al. [1]. The models used in these studies were kernel-based extreme learning machines, ANN, and various ensemble models.

3 USE CASE

The JEMS waste-to-fuel plant produces synthetic diesel (SynDi) from any hydrocarbon-based waste, such as wood, biomass, paper, waste fuels and oils, plastics, textiles, rubber, and agricultural residues. The plant uses a chemical-catalytic de-polymerization process, the advantage of which is that the temperature is too low to produce carcinogenic gasses. It operates continuously and produces about 150 liters of fuel per hour. Although it uses the latest software available and allows remote control, there is no anomaly detection, prediction, or optimization. As a result, there is a great need for better understanding, optimization, and decision-making, given data availability. The company plans to sell and install over 1.500 SynDi systems over the next ten years. In practice, this means many SynDi plants in different locations worldwide.

There are three main chambers in the pipeline, which are named B100, B200, and B300. The plant can be conceptually split into four stages

(1) Feedstock inspecting and feeding;
(2) Drying and mixing (chamber B100);
(3) Processing (chamber B200);
(4) Distilling (chamber B300).

Since there are no sensors in the feedstock inspecting and feeding stage, we focused on the later stages, each of which takes place in one of the main chambers.

In the drying and mixing stage (B100), the starting material is mixed with process oil, lime, and catalyst and is heated. During mixing, the material is broken down into smaller particles, and the water is evaporated. The primary chemical reaction occurs in the processing stage (B200). The material is fed to a turbine, and the reaction product evaporates through the diesel distillation column. If the diesel obtained is not of sufficient quality, it is redistilled in the second distillation stage (B300).

Currently, the plants are operated with highly skilled personnel and high costs for personnel training. Implementing automation, remote control, optimization, and interconnection among the plants would greatly facilitate their operation. Therefore, the main challenge to be solved by integrating AI is the self-control of the chemical process and the plant itself by minimizing the human resources required to operate the plants. Furthermore, operating many SynDi plants also means a significant challenge for ensuring remote control for troubleshooting, maintenance, and repair. AI integration aims to minimize the workforce required to operate the plants, minimize the resulting downtime due to human interaction, enable self-control and predictive maintenance of the SynDi plants, and achieve less downtime and higher production efficiency.

In modeling the waste-to-fuel processes, we decided to model each chamber separately. No model was developed for chamber B300 because it was not active during the period for which we obtained the data. As described above, a second distillation of the fuel is performed in chamber B300 only if the fuel in chamber B200 is not pure enough.

4 METHODOLOGY

4.1 Data analysis

The sensor measurements are from the experimental JEMS plant, which is located in Canada. The data consists of 154 sensors from January 2016 to January 2017. The measurements are taken at one-minute intervals and mostly measure temperature or pressure, but there are also sensors for motor current and valve position, among others. Since the data is from the prototype version of the waste-to-fuel plant, it contains many missing values. Our data set contained an average of 61.607 data points per sensor. We discarded all sensors with less than 6.000 data points and kept only those that corresponded to chambers B100 and B200, giving us data from 39 sensors.

Analysis of the dataset we received revealed that many values were missing. In particular, we noted that there were day-long intervals with a tiny number of measurements. We also noticed that specific sensor values remained constant at low temperatures - a condition best described by the waste-to-fuel plant's inactivity. We, therefore, decided to remove such values. Because there were many ten-minute gaps, we decided to resample the data at fifteen-minute intervals, taking the last value of each interval and assuming that conditions had not changed in the short time since the last measurement - a reasonable assumption for sensor values. The resulting data set contained an average of 7.884 data points per sensor.

We divided the dataset into a train and a test dataset, split on October 31st 2016. The resulting train set included a total of 11.000 samples, and the test set included 3.000 samples.

4.2 Model training

In this research, we compare models that we develop using two different approaches. We first tried the neural network approach, in which the model makes predictions based only on sensor readings from the last five hours. Since the model did not perform better than the baseline, we began the second approach, developing features to describe the time series and capture its patterns. We used linear regression and gradient-boosted tree regressor. All the developed models were compared with the last-value model, which we used as a benchmark.

4.2.1 Neural network approach. We used the model developed for forecasting Tüpras' sensor values. Tüpras is an oil refinery, which is very similar to the JEMS use case. The model was used to forecast sensor values in different units of LPG production. Some of Tüpras' units are distillation columns, similar to JEMS' chamber B200. The model takes only past sensor values as input and predicts values for the future together with the prediction interval. More specifically, it predicts 10th, 50th and 90th percentile, which is the case in all our models that give prediction interval.

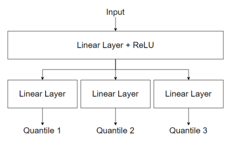


Figure 1: Architecture of the neural network model, which gives the prediction interval.

Figure 1 shows the architecture of the neural network. The model is a feedforward neural network with two layers. First, there is a linear layer with ReLU activation. The second layer has a separate linear layer for each quantile. The hidden dimension of the model is calculated from the number of features and the number of targets using the formula $\lfloor \frac{n_{\text{features}}}{2} \rfloor + n_{\text{targets}}$.

During training, we used the quantile loss function, which is defined as

$$\max\left\{q\cdot\left(y_{\text{true}}-y_{\text{pred}}\right),(1-q)\cdot\left(y_{\text{pred}}-y_{\text{true}}\right)\right\},$$

where q is the observed quantile (in our case, it can be 0.1, 0.5 or 0.9), y_{true} is the true target value and y_{pred} is the corresponding quantile of the prediction. In the case of $q = 0.5$, the loss is equal to the mean absolute error divided by two. When calculating the loss of 10^{th} percentile ($q = 0.1$), a prediction that is greater than the true value is heavily penalized, while a prediction that is lower than the true value has a smaller loss and is therefore encouraged.

The model is implemented in the PyTorch library [7]. Since sensors measure different quantities, the values have to be scaled before learning. Here we used Min-Max scaler from the scikit-learn library, scaling all values between zero and one.

4.2.2 Feature engineering. The neural network model described above did not outperform the benchmark model. As a result, we decided to try another approach, where we developed features that better describe past sensor values and capture their patterns. One of the problems of the neural network model was that it had too many features. We decided to build a separate model for each sensor to tackle this problem. Each model uses only features calculated from the values of the sensor being predicted.

With the help of plant operators, we decided to consider at most five hours of data before the prediction point to issue a forecast. Since the latest data is usually more important in determining future sensor values, we created features on seven different time windows: 30, 45, 75, 120, 180, 240, and 300 minutes. For each time window, we computed the following features:

- average sensor value,
- fraction of peaks in the window,
- percentage change between first and last value in the time window,
- slope (coefficient of the least squares line through the points in the window),
- simple prediction (extension of the least squares line to the future),
- slope ratio (slope on the smaller window divided by the slope on the bigger window).

Besides features mentioned above, which depend on the window size, we also included features that were calculated only on the biggest time window (300 minutes):

- last value,
- maximal value,
- last value relative to the maximal value.

The features above attempt to capture different time series characteristics:

- *trend*: described by percentage change and slope;
- *growth pattern*: described by the fraction of peaks, which indicate whether the growth is steady or it has ups-and-downs. Furthermore, the slope indicates how aggressive such growth is;
- *expected value*: an approximation of the expected value is given through the average, last value, maximal value, and simple prediction.

Using developed features, we trained a linear regression model, and a gradient boosted tree regressor from the CatBoost library [4]. We used root mean squared error (RMSE) for the loss function.

5 RESULTS AND ANALYSIS

We built models for main chambers B100 and B200 with two forecasting horizons (30 and 60 minutes). Tables 1 and 2 show mean squared error (MSE) and mean absolute error (MAE) on chambers B100 and B200, respectively. There are three different neural network models (NN), which differ in the size of the window from which it gets the data.

	horizon = 30min		horizon = 60min	
	MSE	MAE	MSE	MAE
last-value model	21.0533	**1.4320**	50.6636	2.5128
NN, window = 5h	21.7525	1.6512	47.0545	2.5413
NN, window = 3h	19.7441	1.6109	45.3450	2.4127
NN, window = 2h	18.9717	1.6023	46.5047	2.5357
Linear regression	19.4264	1.4634	49.2268	2.5145
Catboost	**16.9030**	1.4478	**38.3066**	**2.3164**

Table 1: MSE and MAE on the test set of models when predicting for chamber B100.

	horizon = 30min		horizon = 60min	
	MSE	MAE	MSE	MAE
last-value model	52.3380	**2.0577**	124.9735	**3.3768**
NN, window = 5h	69.4678	3.8227	129.0330	4.9927
NN, window = 3h	57.9902	3.3601	121.1315	4.7431
NN, window = 2h	55.8769	3.1797	117.4154	4.7146
Linear regression	55.0218	3.2293	115.7457	4.5888
Catboost	**49.3329**	2.5305	**109.5303**	3.9745

Table 2: MSE and MAE on the test set of models when predicting for chamber B200.

From the tables 1 and 2 we can see that the five-hour window's neural network performed worse than the benchmark. The main reason for such poor results was too many features for the amount of data that we have. More precisely, the neural network model uses the values of all sensors in the chamber we are predicting. This means that there are six hundred features resulting in more than two hundred thousand trainable parameters for the model of chamber B200. We also have to consider that the neural network predicts future sensor values and prediction intervals. Therefore, there are too many features and target values for the amount of data that we have.

We included results of two more neural network models with three hours and two-hour windows since reduced window size results in a smaller number of features and trainable parameters. For example, the neural network model with a two-hour time window for chamber B200 had two hundred and forty features and almost fifty thousand trainable parameters. Neural network models with smaller window sizes performed better, which confirms that we had too many features.

The features that we developed using the second approach were used with two models, linear regression and the Catboost model. Comparing those two models, the Catboost model performed better because it can capture more than just linear relationships between the features and the target. The Catboost model also outperformed the neural networks, where one of the main differences is that the neural network uses all sensors from the chamber while the Catboost model uses only sensor values of the sensor which is being predicted. This results in forty-five features for the model that predicts one sensor, which solves the problem of too many features. In addition, the Catboost model produced better results than the benchmark when comparing the mean squared error (MSE). During the training, we used RMSE as a loss function, meaning that RMSE was minimized and, therefore, also MSE.

The tables show that although most models outperform the benchmark regarding MSE, almost all of them do not surpass the benchmark when considering MAE. When measuring MSE, predictions with strong spikes where such spikes do not take place are penalized more. Therefore, models with a competitive MSE are considered to rarely predict spikes when such spikes do not take place. This is a key feature for our use case, given that we are interested to understand whether an irregularity will take place or not. Therefore, the models give valuable information even though the average prediction is not entirely accurate.

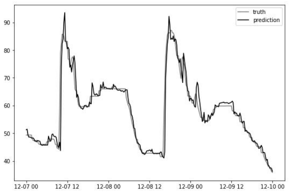


Figure 2: True value and prediction of the Catboost model for a temperature sensor in chamber B100.

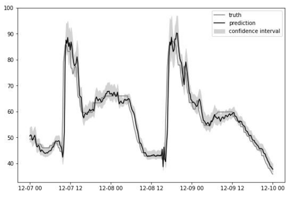


Figure 3: True value and prediction with a confidence interval of the neural network model with a two-hour window for a temperature sensor in chamber B100.

Figure 2 shows the Catboost model prediction on the test set together with the true values of the temperature sensor in chamber B100. The neural network model's prediction of the same sensor is presented in Figure 3. Since the neural network model also outputs prediction interval, it is shown in the abovementioned Figure.

From the plots, we can see that both models can closely predict future sensor values. In the case of the neural network model, the actual value is mainly inside the predicted confidence interval, except when there is a significant change in the sensor value. However, there is no problem with models not being able to predict significant changes resulting from a manual change in plant setpoint parameters, which our data does not capture. Overall, we consider the best model was the Catboost model, given in all cases it outperformed the rest of the models when considering MSE, and also achieved the best MAE when predicting chamber B100 with a time horizon of 60 minutes.

6 CONCLUSION

We compared a set of models to predict sensor values for a waste-to-fuel plant: a neural network, linear regression, gradient-boosted tree regressor, and the last-value model. The last-value model was used as a benchmark. We developed three neural network models which were different in time window size. The neural network models were built based on the hypothesis that a simple neural network and raw sensor readings as features are enough to model the process. The results showed that this is not the case because the process is too complicated for the amount of data that we obtained. Lastly, we used feature engineering to develop features that better describe the time series. Features were used for learning linear regression, and the gradient boosted tree regressor, where the latter produced the best results in our case.

ACKNOWLEDGMENTS

This work was supported by the Slovenian Research Agency and the European Union's Horizon 2020 program project FACTLOG under grant agreement number H2020-869951.

REFERENCES

[1] Walid Kamal Abdelbasset, Safaa M Elkholi, Maria Jade Catalan Opulencia, Tazeddinova Diana, Chia-Hung Su, May Alashwal, Mohammed Zwawi, Mohammed Algarni, Anas Abdelrahman, and Hoang Chinh Nguyen. 2022. Development of multiple machine-learning computational techniques for optimization of heterogenous catalytic biodiesel production from waste vegetable oil. *Arabian Journal of Chemistry* 15, 6 (2022), 103843.

[2] Mortaza Aghbashlo, Wanxi Peng, Meisam Tabatabaei, Soteris A Kalogirou, Salman Soltanian, Homa Hosseinzadeh-Bandbafha, Omid Mahian, and Su Shiung Lam. 2021. Machine learning technology in biodiesel research: A review. *Progress in Energy and Combustion Science* 85 (2021), 100904.

[3] Hemal Chowdhury, Tamal Chowdhury, Pranta Barua, Salman Rahman, Nazia Hossain, and Anish Khan. 2021. *Biofuel production from food waste biomass and application of machine learning for process management.* 96–117. https://doi.org/10.1016/B978-0-12-823139-5.00004-6

[4] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. 2018. CatBoost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363* (2018).

[5] Bidhya Kunwar, HN Cheng, Sriram R Chandrashekaran, and Brajendra K Sharma. 2016. Plastics to fuel: a review. *Renewable and Sustainable Energy Reviews* 54 (2016), 421–428.

[6] F Kusumo, AS Silitonga, HH Masjuki, Hwai Chyuan Ong, J Siswantoro, and TMI Mahlia. 2017. Optimization of transesterification process for Ceiba pentandra oil: A comparative study between kernel-based extreme learning machine and artificial neural networks. *Energy* 134 (2017), 24–34.

[7] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[8] Jilu Lizy Stephen and Balasubramanian Periyasamy. 2018. Innovative developments in biofuels production from organic waste materials: a review. *Fuel* 214 (2018), 623–633.

Machine Beats Machine: Machine Learning Models to Defend Against Adversarial Attacks.

Jože M. Rožanec*
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia
joze.rozanec@ijs.si

Dimitrios Papamartzivanos
Ubitech Ltd
Chalandri, Athens, Greece
dpapamartz@ubitech.eu

Entso Veliou
Department of Informatics and Computer Engineering, University of West Attica
Athens, Greece
eveliou@uniwa.gr

Theodora Anastasiou
Ubitech Ltd
Chalandri, Athens, Greece
tanastasiou@ubitech.eu

Jelle Keizer
Philips Consumer Lifestyle BV
Drachten, The Neatherlands
jelle.keizer@philips.com

Blaž Fortuna
Qlector d.o.o.
Ljubljana, Slovenia
blaz.fortuna@qlector.com

Dunja Mladenić
Jožef Stefan Institute
Ljubljana, Slovenia
dunja.mladenic@ijs.si

ABSTRACT

We propose using a two-layered deployment of machine learning models to prevent adversarial attacks. The first layer determines whether the data was tampered, while the second layer solves a domain-specific problem. We explore three sets of features and three dataset variations to train machine learning models. Our results show clustering algorithms achieved promising results. In particular, we consider the best results were obtained by applying the DBSCAN algorithm to the structured structural similarity index measure computed between the images and a white reference image.

CCS CONCEPTS

• **Information systems → Data mining**; • **Computing methodologies → Computer vision problems**; • **Applied computing**;

KEYWORDS

Cybersecurity, Adversarial Attacks, Machine Learning, Automated Visual Inspection

ACM Reference Format:
Jože M. Rožanec, Dimitrios Papamartzivanos, Entso Veliou, Theodora Anastasiou, Jelle Keizer, Blaž Fortuna, and Dunja Mladenić. 2021. Machine Beats Machine: Machine Learning Models to Defend Against Adversarial Attacks.. In *Ljubljana '22: Slovenian KDD Conference on Data Mining and Data Warehouses, October, 2022, Ljubljana, Slovenia.* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

Artificial Intelligence (AI) solutions have penetrated the Industry 4.0 domain by revolutionizing the rigid production lines enabling innovative functionalities like mass customization, predictive maintenance, zero defect manufacturing, and digital twins. However, AI-fuelled manufacturing floors involve many interactions between the AI systems and other legacy Information and Communications Technology (ICT) systems, generating a new territory for malevolent actors to conquer. Hence, the threat landscape of Industry 4.0 is expanded unpredictably if we also consider the emergence of adversary tactics and techniques against AI systems and the constantly increasing number of reports of Machine Learning (ML) systems abuses based on real-world observations. In this context, Adversarial Machine Learning (AML) has become a significant concern in adopting AI technologies for critical applications, and it has already been identified as a barrier in multiple application domains. AML is a class of data manipulation techniques that cause changes in the behavior of AI algorithms while usually going unnoticed by humans. Suspicious objects misclassification in airport control systems [7], abuse of autonomous vehicles navigation systems [11], tricking of healthcare image analysis systems for classifying a benign tumor as malignant [15], abnormal robotic navigation control [23] are only a few examples of AI models' compromise that advocate the need for the investigation and development of robust defense solutions.

Recently, the evident challenges posed by AML have attracted the attention of the research community, the industry 4.0, and the manufacturing domains [20], as possible security issues on AI systems can pose a threat to systems reliability, productivity, and safety [2]. In this reality, defenders should not be just passive spectators, as there is a pressing need for robustifying AI systems to hold against the perils of adversarial attacks. New methods are needed to safeguard AI systems and sanitize the ML data pipelines from the potential injection of adversarial data samples due to poisoning and evasion attacks.

We developed a machine learning model to address the abovementioned challenges, detecting whether the incoming images are adversarially altered. This enables a two-layered deployment of machine learning models that can be used to prevent adversarial attacks (see Fig. 1): (a) the first layer with models determining whether the data was tampered, and (b) a second layer that operates with regular machine learning models developed to solve particular domain-specific problems. We demonstrate our approach in a real-world use case from *Philips Consumer Lifestyle BV*. This paper explores a diverse set of features and machine learning models to detect whether the images have been tampered for malicious purposes.

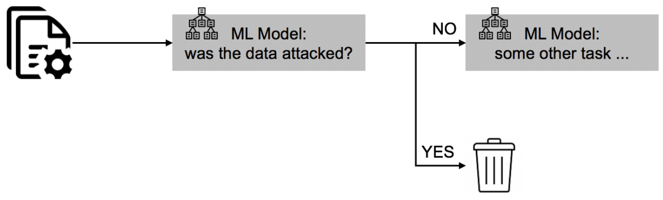


Figure 1: Two-layered deployment of machine learning models can be used to prevent adversarial attacks.

This paper is organized as follows. Section 2 outlines the current state of the art and related works, Section 3 describes the use case, and Section 4 provides a detailed description of the methodology and experiments. Finally, Section 5 outlines the results obtained, while Section 6 concludes and describes future work.

2 RELATED WORK

AML attacks are considered a severe threat to AI systems, and, that is, the research community seeks new robust defensive methods. Image classifiers, those analyzed in this work, are the focal point of the vast majority of the AML literature, as those have been proved prone to noise perturbations. According to the literature, prominent solutions focus on denoising the image classifiers, training the target model with adversarial examples, known as adversarial training, or applying standalone defense algorithms.

Yan et. al. [21] proposed a new adversarial attack called Observation-based Zero-mean Attack, and they evaluated the robustness of various deep image denoisers. They followed an adversarial training strategy and effectively removed various synthetic and adversarial noises from data. In [17], pre-processing data defenses for image denoising are evaluated, highlighting the advantages of such approaches that do not require the retraining of the classifiers, which is a computationally intense task in computer vision.

However, the robustness of adversarial training via data augmentation and distillation is advocated by the majority of the works in the domain. Specifically, Bortsova et al. [3] have focused on adversarial black-box settings, assuming that the attacker does not have full access to the target model as a more realistic scenario. They tuned their testbed to ensure minimal visual perceptibility of the attacks. The applied adversarial training dramatically decreased the performance of the designed attack. Hashemi and Mozaffari [8] trained CNNs with perturbed samples manipulated by various transformations and contaminated by different noises to foster robustness using adversarial training.

On top of the above, several standalone solutions have been proposed. CARAMEL system in [13] offered a set of detection techniques to combat security risks in automotive systems with embedded camera sensors. Hybrid approaches and more general alternatives intrinsically improve the robustness of AI models. A defensive Distillation mechanism against evasion attacks is proposed in [16] being able to reduce the effectiveness of adversarial sample creation from 95% to less than 0.5% on a studied DNN. Subset Scanning was presented in [19] to give the ability to DNNs to recognize out-of-distribution samples.

3 USE CASE

The Philips factory in Drachten, the Netherlands, is an advanced factory for mass manufacturing consumer goods (e.g., shavers, OneBlade, baby bottles, and soothers). Current production lines are often tailored for the mass production of one product or product series in the most efficient way. However, the manufacturing landscape is changing due to global shortages, manufacturing assets and components are becoming scarcer [1], and a shift in market demand requires the production of smaller batches more often. To adhere to these changes, production flexibility, re-use of assets, and a reduction of reconfiguration times are becoming more critical for the cost-efficient production of consumer goods. One of the topics currently investigated within Philips is quickly setting up automated quality inspections to make reconfiguring quality control systems faster and easier. Next to working on the technical challenges of doing this, safety and cyber-security topics are explored, aiming to implement AI-enabled automated quality systems with state-of-the-art defenses, the latter of which is the focus point discussed in this paper.

The dataset used contains images of the decorative part of a Philips shaver. This product is mass-produced and important for the visual appearance of the shavers. Next to that, the part is very close to or in direct contact with the user's skin, where any deviations in its quality could impact shaver performance or even shaver safety. The dataset contains 1.194 images classified into two classes: (a) attacked with the Projected Gradient Descent attack [5], and (b) not attacked.

4 METHODOLOGY

We framed adversarial attack detection as a classification problem. We experimented with three kinds of features: (a) image embeddings (obtained from the Average Pooling Layer of a pre-trained ResNet-18 model ([9])), (b) histograms reflecting grayscale pixel frequencies (with pixel values extending between zero and 255), and (c) structural similarity index measure (SSIM) computed against a white image. While the embeddings provide information about the image as a whole, we considered the histograms and SSIM metric could be useful given the apparent difference between the original and perturbed images. Furthermore, we computed the features across three different datasets (see Fig. 2 for sample images): (a) original set of images, (b) images cropped considering an image slice extending from top to bottom (coordinates (160, 0, 200, 369) - we name this dataset set "Cropped (v1)"), and (c) images cropped

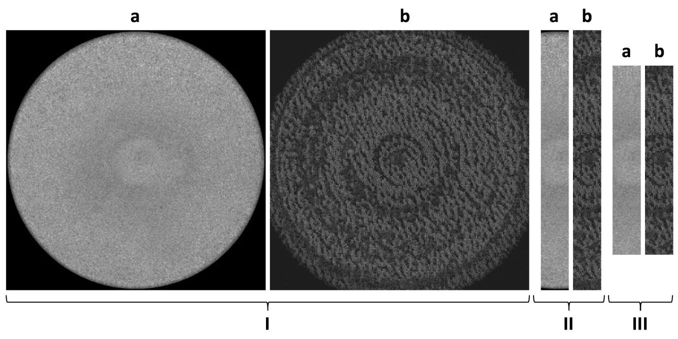


Figure 2: Three sets of images: (a) indicates the original image, while (b) indicates the images attacked with the Projected Gradient Descent attack. The subsets I, II, and III indicate (I) the whole image, (II) cropped image (v1 (considering coordinates (160, 0, 200, 369))), and cropped image (v2 - (considering coordinates (160, 50, 200, 319))).

considering a slice of the central part of the image (coordinates (160, 50, 200, 319) - - we name this dataset set "Cropped (v2)"). By comparing the original image dataset against those obtained by slicing the central part, we sought to understand if the models' predictive power increased by looking at a specific area of the image rather than the whole.

We first trained three machine learning models: Catboost [18] with Focal Loss [14] (trained over 150 iterations, and considering a tree depth of ten, while evaluating the performance during training with the logloss metric), Logistic Regression (the dataset was scaled between zero and one, considering the train set, and transformed to ensure zero mean and unit variance), and KMeans (the dataset was transformed to ensure zero mean and unit variance, and the model initiated with random initialization and seeking to generate two clusters). We evaluated our experiments with a ten-fold stratified cross-validation ([12, 22]), using one fold for testing and the rest of the folds to train the model. Furthermore, to avoid overfitting, we performed a feature selection using the mutual information to evaluate the most relevant ones and select the *top K* features, with $K = \sqrt{N}$, considering N to be equal to the number of data instances in the train set [10]. Finally, we measured our models' performance with a custom metric (DP_{AUCROC}) that summarizes the discriminative power as computed from the area under the receiver operating characteristic curve (AUC ROC, see [4]) (see Eq. 1). The metric ranges from zero (no discriminative power) to one (perfect discriminative power) and it preserves the AUC ROC desirable properties of being threshold independent and invariant to *a priori* class probabilities.

$$DP_{AUC_ROC} = 2 \cdot |(0.5 - AUCROC)| \qquad (1)$$

Based on the good results obtained in the clustering setting, we decided to conduct additional experiments, running the DBSCAN algorithm [6] over all existing data. The advantage of such an algorithm is that it can estimate the clusters with no prior information regarding the number of expected clusters. Therefore, if working well, it would be useful to generalize the approach toward detecting new cyberattacks where no labeled data exists yet. We consider such a characteristic to be fundamental to production environments. For the models resulting from the three abovementioned datasets, we measured the estimated number of clusters, estimated number of noise points, homogeneity (whether the clusters contain only samples belonging to a single class), completeness (whether all the data points members of a given class are elements of the same cluster), V-measure (harmonic mean between homogeneity and completeness), adjusted Rand index (similarity between clusterings obtained by the proposed and random models), and the Silhouette Coefficient (estimates the separation distance between the resulting clusters). We ran the DBSCAN algorithm measuring the distance between clusters with the Euclidean distance, considering the maximum distance between two samples for one to be considered as in the neighborhood of the other to be 0,3. Furthermore, we considered that at least ten samples in a neighborhood were required for a point to be considered as a core point.

5 RESULTS AND ANALYSIS

Model		Catboost	KMeans	Logistic regression
Embeddings	**Original image**	0.0167	**1.0000**	*0.0228*
	Cropped (v1)	*0.0014*	**1.0000**	0.0003
	Cropped (v2)	0.0181	**1.0000**	*0.0213*
SSIM	**Original image**	0.0152	**1.0000**	*0.0184*
	Cropped (v1)	*0.0008*	**1.0000**	0.0004
	Cropped (v2)	0.0179	**1.0000**	*0.0195*
Histograms	**Original image**	0.0016	**1.0000**	*0.0030*
	Cropped (v1)	0.0003	**1.0000**	*0.0011*
	Cropped (v2)	0.0018	**1.0000**	*0.0031*

Table 1: Results obtained across classification experiments. We measure models' performance with Eq. 1. Best results are bolded, *second-best are italicized.*

We present the results obtained in our classification experiments in Table 1. We found the KMeans models achieved perfect discrimination in all cases, while the second-best model was the Logistic regression, which had second-best results in all but two cases. Nevertheless, the Logistic regression and the Catboost models achieved a low discriminative power, almost unable to distinguish between tampered and non-tampered images. Regarding the features, we found that the best average performance was obtained when training the models on the *Cropped (v2)* dataset, followed by those trained on the whole images.

When running the DBSCAN algorithm (see results in Table 2), we found the best results were obtained considering the SSIM measure. Furthermore, using the SSIM issued excellent results in all cases. The best ones were obtained considering the *Cropped (v1) dataset*, while the second-best was achieved with the *Cropped (v2) dataset*. Using the SSIM only, the DBSCAN algorithm was able to correctly group the instances into two groups and misclassified at most a single instance. However, the performance achieved either with embeddings or histograms was not satisfactory. When considering histogram features, the DBSCAN algorithm was not able to discriminate between instances, creating a single cluster. On the other hand, when considering embeddings, DBSCAN created three clusters that issued a bad performance, considering most of the

	Embeddings			SSIM			Histograms		
	Original image	Cropped (v1)	Cropped (v2)	Original image	Cropped (v1)	Cropped (v2)	Original image	Cropped (v1)	Cropped (v2)
Number of clusters	3	1	1	**2**	**2**	**2**	1	1	1
Number of noise points	1010	794	887	1	**0**	1	621	603	606
Homogeneity	0.1770	0.4550	0.3170	**1.0000**	**1.0000**	**1.0000**	0.8550	0.9290	0.9150
Completeness	0.2090	0.4940	0.3860	*0.9910*	**1.0000**	*0.9910*	0.8560	0.9290	0.9150
V-measure	0.1920	0.4740	0.3480	*0.9960*	**1.0000**	*0.9960*	0.8550	0.9290	0.9150
Adjusted Rand index	0.0710	0.4350	0.2540	*0.9980*	**1.0000**	*0.9980*	0.9020	0.9600	0.9500
Silhouette coefficient	0.0750	0.4310	0.2660	0.8980	**0.9590**	*0.9070*	0.8330	0.8970	0.8800

Table 2: Results obtained across clustering experiments. Best ones are bolded, *second-best are italicized.*

points to be noisy. We, therefore, conclude that the only promising results were those obtained considering the SSIM. Nevertheless, we consider further research is required to understand whether this kind of feature can be useful across a wide range of attacks and in the real-world. SSIM provides metadata describing the images. Given high-quality attacks aim to reduce the visual footprint on the images, it remains an open question to which extent can the SSIM capture weak footprints and therefore enable similar discriminative capabilities on machine learning models.

6 CONCLUSION

In this work, we explored multiple sets of features and machine learning models to determine whether an image has been tampered with for the purpose of an adversarial attack. While the difference between attacked and non-attacked images is evident to the human eye, it is not to the machine learning algorithms. We found that the Catboost and Logistic regression models could almost not discriminate between both cases. On the other hand, the clustering algorithms (KMeans and DBSCAN) had a stronger performance. While the KMeans models did so perfectly, regardless of the features, the DBSCAN model only performed well using the SSIM. We consider the strength of such a model the fact that no *a priori* information regarding the classes is required, therefore saving the annotation effort and providing greater flexibility towards future adversarial attacks. Our future research will focus on testing a wider range of cyberattacks while ensuring the attack will not be discernable to the human eye.

ACKNOWLEDGMENTS

This work was supported by the Slovenian Research Agency and the European Union's Horizon 2020 program project STAR under grant agreement number H2020-956573.

REFERENCES

[1] [n.d.]. European Economic Forecast. Autumn 2021. https://economy-finance.ec.europa.eu/publications/european-economic-forecast-autumn-2021_en. Accessed: 2022-08-05.

[2] Adrien Bécue, Isabel Praça, and João Gama. 2021. Artificial intelligence, cyber-threats and Industry 4.0: Challenges and opportunities. *Artificial Intelligence Review* 54, 5 (2021), 3849–3886.

[3] Gerda Bortsova, Cristina González-Gonzalo, Suzanne C. Wetstein, Florian Dubost, Ioannis Katramados, Laurens Hogeweg, Bart Liefers, Bram van Ginneken, Josien PW Pluim, and Mitko Veta. 2021. Adversarial attack vulnerability of medical image analysis systems: Unexplored factors. *Medical Image Analysis* 73 (2021), 102141. Publisher: Elsevier.

[4] Andrew P. Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition* 30, 7 (1997), 1145 – 1159. https://doi.org/10.1016/S0031-3203(96)00142-2

[5] Yingpeng Deng and Lina J Karam. 2020. Universal adversarial attack via enhanced projected gradient descent. In *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 1241–1245.

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

[7] Dan-Ioan Gota, Adela Puscasiu, Alexandra Fanca, Honoriu Valean, and Liviu Miclea. 2020. Threat objects detection in airport using machine learning. In *2020 21th International Carpathian Control Conference (ICCC)*. IEEE, 1–6.

[8] Atiyeh Hashemi and Saeed Mozaffari. 2021. CNN adversarial attack mitigation using perturbed samples training. *Multim. Tools Appl.* 80 (2021), 22077–22095.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[10] Jianping Hua, Zixiang Xiong, James Lowey, Edward Suh, and Edward R Dougherty. 2005. Optimal number of features as a function of sample size for various classification rules. *Bioinformatics* 21, 8 (2005), 1509–1515.

[11] A. Kloukiniotis, A. Papandreou, A. Lalos, P. Kapsalas, D.-V. Nguyen, and K. Moustakas. 2022. Countering adversarial attacks on autonomous vehicles using denoising techniques: A Review. *IEEE Open Journal of Intelligent Transportation Systems* (2022). Publisher: IEEE.

[12] Max Kuhn, Kjell Johnson, et al. 2013. *Applied predictive modeling*. Vol. 26. Springer.

[13] Christos Kyrkou, Andreas Papachristodoulou, Andreas Kloukiniotis, Andreas Papandreou, Aris Lalos, Konstantinos Moustakas, and Theocharis Theocharides. 2020. Towards artificial-intelligence-based cybersecurity for robustifying automated driving systems against camera sensor attacks. In *2020 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, 476–481.

[14] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*. 2980–2988.

[15] Xingjun Ma, Yuhao Niu, Lin Gu, Yisen Wang, Yitian Zhao, James Bailey, and Feng Lu. 2021. Understanding adversarial attacks on deep learning based medical image analysis systems. *Pattern Recognition* 110 (2021), 107332.

[16] Nicolas Papernot, Patrick D. McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2015. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. *CoRR* abs/1511.04508 (2015). arXiv:1511.04508 http://arxiv.org/abs/1511.04508

[17] Marek Pawlicki and Ryszard S. Choraś. 2021. Preprocessing Pipelines including Block-Matching Convolutional Neural Network for Image Denoising to Robustify Deep Reidentification against Evasion Attacks. *Entropy* 23, 10 (2021), 1304. Publisher: MDPI.

[18] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems* 31 (2018).

[19] Skyler Speakman, Srihari Sridharan, Sekou Remy, Komminist Weldemariam, and Edward McFowland. 2018. Subset scanning over neural network activations. *arXiv preprint arXiv:1810.08676* (2018).

[20] Entso Veliou, Dimitrios Papamartzivanos, Sofia Anna Menesidou, Panagiotis Gouvas, and Thanassis Giannetsos. 2021. *Artificial Intelligence and Secure Manufacturing: Filling Gaps in Making Industrial Environments Safer.* Now Publishers. 30–51 pages. https://doi.org/10.1561/9781680838770.ch2

[21] Hanshu Yan, Jingfeng Zhang, Jiashi Feng, Masashi Sugiyama, and Vincent YF Tan. 2022. Towards Adversarially Robust Deep Image Denoising. *arXiv preprint arXiv:2201.04397* (2022).

[22] Xinchuan Zeng and Tony R Martinez. 2000. Distribution-balanced stratified cross-validation for accuracy estimation. *Journal of Experimental & Theoretical Artificial Intelligence* 12, 1 (2000), 1–12.

[23] Fangyi Zhang, Jürgen Leitner, Michael Milford, Ben Upcroft, and Peter Corke. 2015. Towards vision-based deep reinforcement learning for robotic motion control. *arXiv preprint arXiv:1511.03791* (2015).

Addressing climate change preparedness from a smart water perspective

Alenka Guček *, Joao Pita Costa ** ***, M.Besher Massri * ** *******, João Santos Costa *, Maurizio Rossi ****, Ignacio Casals del Busto *****, Iulian Mocanu ******

* Institute Jozef Stefan, Slovenia, ** IRCAI, Slovenia, *** Quintelligence, Slovenia, **** Ville de Carouge, Switzerland, ***** Aguas de Alicante, Spain, ****** Apa Braila, Romania, ******* Jozef Stefan International Postgraduate School

ABSTRACT

Observing the world on a global scale can help us understand better the role of water and water resource management utilities in a climate change context that engage us all. The usage of machine learning algorithms on open data measurements and statistical indicators can help us understand the behavioral changes in seasons and better prepare. These are complemented by powerful text mining algorithms that mine worldwide news, social media, published research and patented innovation towards best practices from success stories. In this paper, we propose a data-driven global observatory that puts together the different perspectives of media, science, statistics and sensing over heterogeneous data sources and text mining algorithms. We also discuss the implementation of this global observatory in the context of epidemic intelligence, monitoring the impact of climate change, and the value of this global solution in local contexts and priorities.

CCS CONCEPTS

• Real-time systems • Data management systems • Life and medical science

KEYWORDS Climate Change Preparedness, Data-driven Decision-making, Water Resource Management, Smart Water, Observatory, Water Digital Twin, Deep Learning, Text Mining, Interactive Data Visualization

1 Introduction

In the present decade, Climate Change has become positioned as one of the world priorities, a global problem with great socio-economic impact. It has been in the focus of European and Worldwide strategies, rapidly changing priorities towards sustainability and environmental efficiency, transversely to most domains of action. The European Commission's Green Deal [5] is a good example of this, aiming for a climate neutral Europe in 2050, and boosting the economy through green technology over a new framework to understand and position water resource management in the context of the challenges of tomorrow [1]. In the context of the NAIADES project [3] we repurpose and customize the NAIADES Water Observatory, adding a measurements dimension to its text mining capabilities to allow for forecasts on, e.g., water level and temperature to complete the perspective on the impact of climate change for the preparedness both of water management utilities and users as in, e.g., smart agriculture. This will improve the climate change preparedness of water resource management facilities and local authorities in a global context, in particular in European regions where water scarcity or extreme weather events are predicted. The water-related climate change topics that we are already addressing include, e.g., water reuse, wastewater management, saline intrusion and groundwater contamination.

In this paper we will discuss our contribution to this cause, through the NAIADES Water Observatory (accessible at naiades.ijs.si) [12], focusing on water-related aspects, allowing the user to explore a combination of perspectives offered over layers of information sourced in statistics, historical measurements, multilingual news and social media to published science, weather models and indicators. It is also being used in the context of extreme weather events to analyze worldwide trends and best practices in water topics like, e.g., floods, landslide, and contamination [9], building business intelligence from the available open data in combination with data streams [11].

The NAIADES Water Observatory is not only contributing to the improvement of European sustainability in water-related activities and business intelligence but it is also providing an active role to local actors in improving together with municipalities and water resource management utilities the efficient use of resources [13]. This local perspective is especially important for providing information at the local granularity, which enables communities or municipalities to build solutions that are relevant for their specific cases.

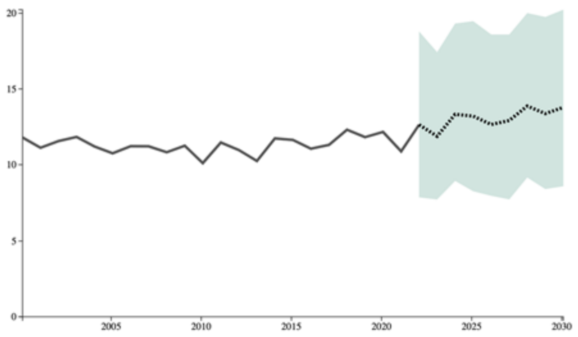


Figure 1: **Long-term forecast of 10 years (average per year) built on 20 years of data to understand the behavior of air temperature, water levels and temperature and the consequent changes within seasons.**

2 Understanding behaviors from data

In the era of Big Data where technologies and sensors are every day cheaper and more efficient, a wide range of useful measurements is available and can be used to forecast weather and water resource behaviors and to identify environmental trends with local granularity.

With the motivation to grasp a realistic perspective on the impact of climate change in the region of Carouge, Switzerland, we obtained 20 years of water levels and water temperature data (sourced by Meteoswiss Data Portal IDAWEB), and we were able to build a 10-year forecast that allows us to see a signal of the global trend.

For this aim, we have developed a Long Short Term Memory (LSTM) neural network, which is a type of Recurrent Neural Network, widely used for predicting sequential data. In order to optimize the performance and accuracy of the LSTM, we used some results from Differential Geometry and Chaos Theory such as Takens’ Embedding Theorem, Shannon Entropy, Conditional Shannon Entropy, Markov Chains, etc. This theoretical support was key for obtaining the optimal number of timesteps [4] and to produce a long-term forecast aiming to observe the weather behavior across the historical data collected and a perspective on the future seasons based on the derived prediction, represented by the three parameters - temperature, humidity and rainfall - or the water levels in rivers, lakes and basins in the area determined by the geolocation provided by the NAIADES use cases. The time series of historical data in Figure 1 indicates that already the air temperature yearly averages are increasing, and this increase is predicted also for the next 10 years. Comparing our model with the Meteoswiss model for the area, the differences were minimal. To emphasize the changes throughout the year, we added a per year visualization (Figure 2), where one can compare the seasonal trends for the local weather and water parameters.

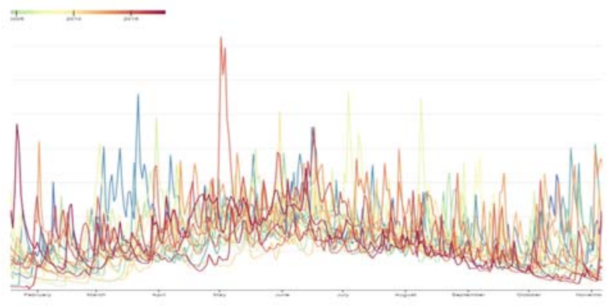


Figure 2: **The weather across seasons over the past 20 years distinguished by seasons, exhibiting high temperature periods earlier in the year.**

To further explore the relations of multivariate timeseries data, we have developed the State analysis tool [14]. With this technology we automatically abstract data as states of the Markov chain and transitions between them. This allows for ingestion of large datasets, and due to hierarchical clustering the data can be observed on several levels. This tool works especially well for observing long term behavior and exposing recurrent patterns. In the context of climate change preparedness, the aim was to better understand the reality of the seasons as defined by the weather parameters as well as the water level and temperature over the past 20 years. Depicted in Figure 3 are the transitions between seven states we can already depict in the municipality of Carouge, Switzerland and the surrounding area. Five of those states correspond to a passage between Spring-Summer and Summer-Autumn, and to Summer itself, characterized by the states indicating a high water temperature. With the impact of climate change in redefining seasons this tool can help to plan ahead, having in mind the granularity of the data that can be customized to predefined geographic regions where relevant water resources are located.

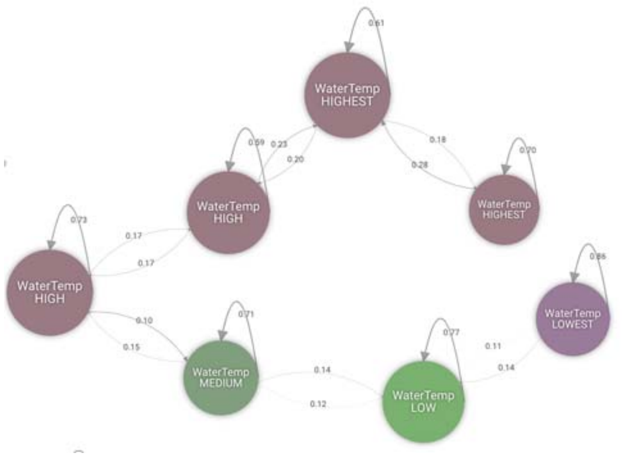


Figure 3: **The analysis of the impact of climate change on water levels and temperature across seasons using Markov chains**

3 Enrichment with local indicators

Water is fundamental to all human activity and ecosystem health, and is a topic of rising awareness in the context of climate change. Water resource management is central to those concerns, with the industry accounting for over 19% of global water withdrawal, and agricultural supply chains are responsible for 70% of water stress [10]. In 2015 the UN established "clean water and sanitation for all" as one of the 17 Sustainable Development Goals, aiming for eight targets to be achieved by 2030 [2].

To exploit the functionality for the customization at the level of local regional providers, news monitoring, and exploration of scientific research can be customized to observed problems, e.g., groundwater contamination. Moreover, ingestion of local indicators can be customized also. These agencies (e.g. Aguas de Alicante) are collecting data on their water resource management services to improve the customer satisfaction and optimize their system, aiming for a smart water [6] approach for the optimization of resources and means, often deploying intelligent systems close to the idea of a water digital twin [7].

Together with the municipality of Carouge, Switzerland, and with the water management utilities of Alicante, Spain, and Braila, Romania, we have collected open data from national data portals and environmental agencies with a regional granularity to be able to assess the comparative progress of regions through the visual data representation of indicators (see Figure 4). Through this interactive data visualization we can investigate the progress on a variety of topics (with three simultaneous parameters represented over a bubble chart) that are much relevant to the analysis of climate change, including water availability, reused and treated water, or water usage by populations and industry. With the appropriate combination of variables in comparison, the user can identify the most efficient regions over the country.

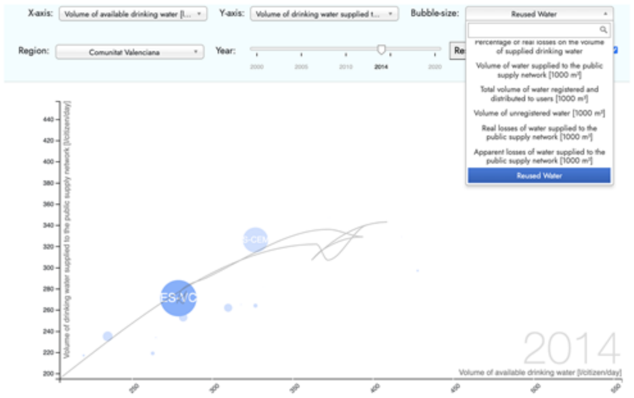

X-axis:
Volume of available drinking water [l...
Y-axis:
Volume of drinking water supplied t...
Bubble-size:
Reused Water
Region:
Comunitat Valenciana
Year:
2000
2005
2010
2014
2020
Percentage of real losses on the volume of supplied drinking water
Volume of water supplied to the public supply network [1000 m³]
Total volume of water registered and distributed to users [1000 m³]
Volume of unregistered water [1000 m³]
Real losses of water supplied to the public supply network [1000 m³]
Apparent losses of water supplied to the public supply network [1000 m³]
Reused Water
Volume of drinking water supplied to the public supply network [l/citizen/day]
ES-VC
2014
Volume of available drinking water [l/citizen/day]

Figure 4: **The comparison of indicators in the Spanish regions across time**

To better understand the comparative progress of each region on the selected water-related topics, we also enable the representation of the time-series curves (see Figure 5) to identify transitions, peaks and other behaviors (per parameter in analysis) that are otherwise not seen in the bubble chart animation.

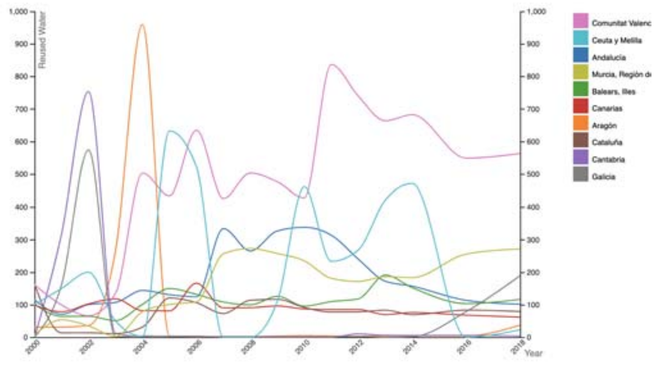


Figure 5: **The curves comparing regional indicators on water topics (as, e.g., reused water in Spain)**

4 Knowledge extracted from news, social media and scientific research

The NAIADES Water Observatory also allows for a news monitoring perspective with global and local coverage on topics like, e.g., water scarcity and water quality. It is particularly relevant in the surrounding regions of the water resource management agencies, but also at a worldwide level recurring to its multilingual capacity to access success stories and best practices form similar scenarios happening worldwide. This is based on the Event Registry news engine [8] that collects over 300 thousand news articles daily in over 60 languages. In the past 3 months we were able to capture almost 33 thousand articles relating both with water and with the climate crisis, 1500 of them happening in Spain and relating to concepts such as, e.g., draught, wildfire heat wave, irrigation and extreme weather.

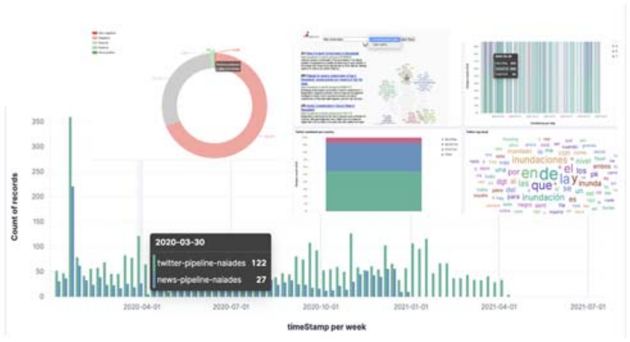


Figure 6: **The combined perspective of multilingual news, social media and scientific research on water scarcity and extreme weather aiming to identify best practices and success stories**

This global system is also capturing the filtered Twitter feed on 10% of the signal, to identify posts related to heat wave and drought (see Figure 6).

The scientific research on climate change topics can bring an important complement in this context, providing success stories and best practices that can be extracted from the textual data, and explored with complex data visualization technology allowing the user to powerful Lucene-based queries over the article's metadata and to relate that research across time suggesting related topics (see Figure 7). These data analytics technologies are able to analyze simultaneously multiple time-series providing interactive exploration tools to understand trends in climate change research and water topics related to it.

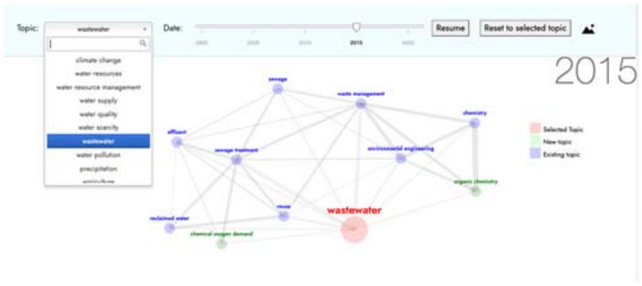


Figure 7: **The trends over time that relate to the topic Climate Change in the scientific literature**

5. Conclusions and further work

Adapting to climate change is an important topic for water management services, since their work is quintessential for the well-being of people. Understanding the seasonality changes and forecasting the availability of resources at the local levels is therefore crucial to enable relevant adaptation at the correct granularity.

Although the predictions are in accordance with IPCC's and Meteoswiss forecasting, this preliminary work needs to be extended with ingesting several other data variables and compared to the existing widely used models to bring more accurate insight specially for the weather data, but also the water-relevant resources.

ACKNOWLEDGMENTS

We thank the support of the European Commission on the H2020 NAIADES project (GA nr. 820985).

REFERENCES

[1] A. Akhmouch, C. Delphine and P. G. Delphine Clavreul. Introducing the OECD principles on water governance. Water International, 43: 5–12, 2018
[2] V. Blazhevska. United Nations launches framework to speed up progress on water and sanitation goal. United Nations Sustainable Development, 2020
[3] CORDIS, "NAIADES Project". [Online]. Available: https://cordis.europa.eu/project/id/820985 [Accessed 1 9 2020].
[4] Costa J., Kenda K., Pita Costa J. (2021). Entropy for Time Series Forecasting. In: Slovenian Data Mining and Data. Warehouses conference (SiKDD2021)
[5] European Commission, "European Green Deal," 2019. [Online]. Available: https://ec.europa.eu/info/strategy/priorities-2019-2024/ european-green-deal_en. [Accessed 1 9 2020].
[6] C. Sun, V. Puig, G. Cembrano. (2020). Real-Time Control of UrbanWater Cycle under Cyber-Physical Systems Framework. Water: 12, 406.
[7] Di Nardo et al. (2018). On-line Measuring Sensors for Smart Water Network Monitoring. EPiC Series in Engineering. 3: 572-581
[8] G. Leban, B. Fortuna, J. Brank and M. Grobelnik, "Event registry: learning about world events from news," Proceedings of the 23rd International Conference on World Wide Web, pp. 107-110, 2014.
[9] M. Mikoš, N. Bezak, J. Pita Costa, M. Besher Massri, M. Jermol, M. Grobelnik (2021) Natural-hazard-related web observatories as a sustainable development tool in Progress, in Landslide Research and Technology, Springer, Vol. 1, No. 1, 2022.
[10] Our World in Data (2022). Water Use Stress. https://ourworldindata.org/water-use-stress. [Accessed 1 8 2022]
[11] J. Pita Costa (2022). Business intelligence built from open data. Water World Magazine. [Online]. Available: https://www.waterworld.com/water-utility-management/smart-water-utility/article/14234325/2203wwint [Accessed 1 8 2022]
[12] J. Pita Costa (2021). Observing water-related events to support decision-making. Smart Water Magazine. [Online]. Available: https://smartwatermagazine.com/news/naiades-project/observing-water-related-events-support-decision-making [Accessed 1 8 2022]
[13] J. Pita Costa, I. Casals del Busto, A. Guček, et al (2022). Building A Water Observatory From Open Data. Proceedings of the IWA 2022.
[14] L. Stopar, P. Škraba, M. Grobelnik, and D. Mladenić (2018). StreamStory: Exploring Multivariate Time Series on Multiple Scales. IEEE transactions on visualization and computer graphics 25. 4: 1788-1802.

SciKit Learn vs Dask vs Apache Spark Benchmarking on the EMINST Dataset

Filip Zevnik, Din Music, Carolina Fortuna, Gregor Cerar
Department of Communication Systems, Jozef Stefan Institute
Ljubljana, Slovenia
zevnikfilip@gmail.com

***Abstract*—As datasets for machine learning tasks can become very large, more consideration to memory and computing resource usage has to be given. As a result, several libraries for parallel processing that improve RAM utilization and speed up computations by parallelizing ML jobs have emerged. While SciKit Learn is the typical go to library for practitioners, Dask is a parallel computing library that can be used with SciKit and Apache Spark is an analytics engine for large-scale data processing that includes some machine learning techniques. In this paper, we benchmark the three solutions for developing ML pipelines with respect to data merging and loading and subsequently for training and predicting on the extended MNIST (eMNIST) dataset under Linux and Windows OS. Our results show that Linux is the better option for all of the benchmarks. For low amounts of data plain SciKit learn is the best option for machine learning, but for more samples, we would choose Apache Spark. On the other hand, when it comes to dataframe manipulation Dask beats a normal pandas import and merge.**

***Index Terms*—Apache Spark, Dask, machine learning, Pandas, import**

I. Introduction

As datasets for machine learning tasks can become very large, more consideration to memory and computing resource usage has to be given. As a result, several libraries for parallel processing that improve RAM utilization and speed up computations by parallelizing ML jobs have emerged. While SciKit Learn [1] is the typical go to library for practitioners, Dask [2] is a parallel computing library that can be used with SciKit to improve memory and CPU utilization. Dask improves memory utilization by not immediately loading all the data, but only pointing to it. Only part of the data is loaded on a per need basis. It also enables using all available cores on a system to train a model. Apache Spark is an analytics engine written in Java and Scala for processing large-scale data that incorporates some machine learning techniques and is tightly integrated with the Spark architecture.

While there are other libraries [3] that enable parallelization of ML, when it comes to distributed computing tools for tabular datasets, Spark and Dask are the most popular choices today. Even though Spark is an older, more stable solution, Dask is part of the vibrant Python ecosystem and both technologies excel at parallelization. While the two solutions have been already been benchmarked on big data pipelines [4] and on various image processing and learning scenarios [5]–[7]. The work in [7] is the closest to this one, however they focused on evaluating the tradeoffs in parellelizing feature extraction and clustering while this work focuses on evaluating data loading and merging and subsequent classification.

This work was funded by the Slovenian Research Agency ARRS under program P-0016.

In this paper, we benchmark the three solutions for developing ML pipelines with respect to data merging and loading and subsequently for training and predicting on the extended MNIST (eMNIST) dataset under Linux and Windows OS. Our results show that Linux is the better option for all of the benchmarks. For low amounts of data plain SciKit learn is the best option for machine learning, but for more samples, we would choose Apache Spark. On the other hand, when it comes to dataframe manipulation Spark is behind Dask, and Dask beats a normal pandas import and merge. The contribution of this paper is the benchmarking of three ML libraries across various data sizes and two operating systems on two parts of the ML model development pipeline.

The remainder of the paper is structured as follows. Section II discusses related work. Section III presents the methodology used in the benchmarking. Section IV evaluates the comparison. Finally, Section V presents our conclusions.

II. Related work

Chintapalli et al. (2016) [8] compared streaming platforms Flink, Storm and Spark. The paper focuses on real-world streaming scenarios using ads and ad campaigns. Each streming platform was used to build a pipeline that identifies relevant events, which were sources from Kafka. In addition, Redis was used for storing windowed count of relevant events per campaign. The test system contained 40 nodes, where each node contained 2 CPUs with 8 cores and 24GB of RAM. All nodes were interconnected using a gigabit ethernet connection. The experiment encompassed Kafka producing events at set rate with 30 minutes interval between each batch was fired. The results showed that both Flink and Storm were almost equal in terms of event latency, while Spark turned out to be the slowest of the three.

Dugré et al. (2019) [4] compared Dask and Spark on the neuroimaging big data pipelines. As neuroimaging requires a large amount of images to be processed, Spark and Dask were in the time of writing the best suited Big Data engines. The paper compares the technologies with three different pipelines. First is incrementation, second is histogram and the final

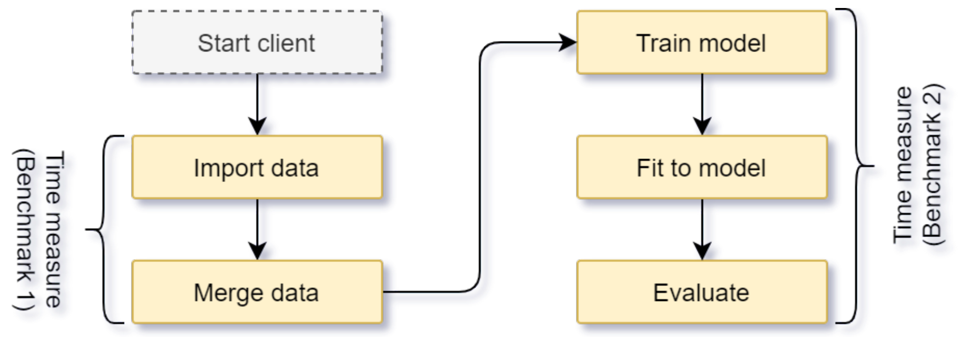


Fig. 1. Workflow of the Machine learning test example used for benchmarking.

one is a BIDS app example (a map-reduce style application). All comparisons were done on BigBrain and CoRR datasets, with sizes of 81GB and 39GB respectively. The authors have concluded that all platforms perform very similarly and that the incrementation of worker nodes is not always the optimal solution due to the transfer times and overall overhead. While all platforms yielded similar results, the Spark is claimed to be the fastest out of the three platforms.

Nguyen et al. (2019) [6] evaluated SciDB, Myria, Spark, Dask and TensorFlow to figure out which system is best suited for image processing. Similarly to [4], the authors compared the systems using different pipelines. For comparison, the authors used 2 datasets, both over 100GB in size. The comparison reveled that Dask and Spark are comparable in the performacnce as well as the ease of use.

Mehta et al. (2016) [5] presented the satellite data processing pipeline. The pipeline consists of two steps, a feature extraction step and a clustering step. The baseline pipeline used the Caffe deep learning library and SciKit. The improved pipeline used Keras along with Spark and Dask for multi-node computation. They found that while Spark was the fastest in terms of computational time required per task, Dask used almost half the memory compared to Spark due to recalculation of the intermediate values. SciKit Learn was not able to complete the task and was excluded from the final comparison. It was concluded that Spark is the best performer, while Dask is the easiest to use.

Cheng et al. (2019) [7] presented a comparison of the RADICAL-Pilot, Dask and Spark for image processing. All three systems were tested using watershed and a blob detector algorithms. Each test was split into two parts, a weak scaling algorithm where the amount of data to be processed was increased alongside the number of nodes, and a strong scaling algorithm where the amount of data stayed the same and the number of nodes increased. The evaluation showed that Dask outperformed Spark on weak scaling, while Spark excelled in the strong scaling part.

III. Methodology

To benchmark the three solutions, namely SciKit learn, Dask and Spark, we single out two parts of the end-to-end model development process depicted in Figure 1. We first time the data importing and merging process, referred to as Benchmark 1 in the figure, followed by model training and evaluation denoted by Benchmark 2. While the time required to train the model is usually the most important metric because it takes up most of the computation time, importing and merging the input data cannot be ignored. As described in Algorithm 1, for Benchmark 1, training data was imported and then merged. For SciKit Learn dataframes were used all along and no parallelization was used while for Dask and Spark parallelization was turned on.

Algorithm 1: Import and merge benchmarking process.

```
Enable parallelization
Require: data_a and data_b
Merge the DataFrames
Convert data to a pandas DataFrame
```

Algorithm 2: Train/fit and evaluate benchmarking process.

```
Enable parallelization
Import and setup data
train = [80% of the samples], test = [20% of the samples]
Define ML algorithm
Fit the data
Predict the samples
Evaluate - F1
```

As described in Algorithm 2, for Benchmark 2 in Figure 1, an example of machine learning with a decision tree classifier depicts the workflow of the machine learning test example. First, parellelization is enabled for Dask and Spark and immediately after that the data is imported and modified accordingly to fit the test scenario. Next, the decision tree classifier is trained using various training data size, dividing the data set into a training subset and a test subset. The training subset represents 80% of the original dataset and for the training subset the remaining data is used, representing 20% of the original dataset. Each task is run with 5 different sample sizes, ranging from 50k to 250k samples, with a step of

50k samples. Finally, the execution report with the calculation times of each task is generated.

To realize these benchmarks[1], we used the extended MNIST or EMNIST dataset[2]. The data set contains approximately 250k samples of handwritten digits, resulting in total size of 516MB. The size of all images is exactly the same, 28 by 28 pixels and each pixel has a value ranging from zero to 255. The dataset is represented in the CSV (Comma Separated Values) format with the first column being the label and the rest of the columns representing 784 pixels. For the benchmarks, different data set sizes, ranging from 50k to 250k samples with a 50k step were generated.

In addition, each data set size was tested on Dask and Spark with 1, 2 and 4 workers. Therefore, the programs used to test computation time on Windows and Linux operating systems have the same complexity. All tests were performed on equivalent Windows and Linux virtual machines running on the 6 CPU core machine with 10 GB of RAM.

IV. Results

In this section we provide the results of the benchmarks collected using the methodology described in Section III.

A. Import and merge

First, we present in Figure 2 the import and merge times for 100k samples on Linux without parallelization across the the three platforms. In the first bar, it can be seen that importing (i.e. loading the data into memory) takes most of the time with Pandas. Merging (i.e. concatenation) is relatively negligible while computation is not relevant in this case as after merging it already returns the desired data structure. The total import and merge time is slightly above 4s.

From the second bar, it can be seen that importing and merging is negligible with Dask as doesn't load anything into memory at these steps, rather it prepares only recipes that will be executed during the most time consuming compute phase. During compute, Dask turns a lazy collection into its in-memory equivalent, in our case, the Dask dataframe turns into a Pandas dataframe. Overall, it can be seen that on a single node, Dask is comparable to Pandas, with a total import and merge time slightly below 4s.

Finally, from the last bar, it can be seen that Spark import and merge are very fast and efficient, taking below 2s. However, when transforming the internal data structure of Spark into pandas (i.e. during the compute phase in this case) is very time consuming. We added this step so that the final outcome is consistent with the other two (i.e. Pandas data structure), however in the end-to-end ML pipeline the ML algorithm will be trained directly using Spark's internal data structure.

Figure 3 shows how the import and merge times fare as a function of worker nodes for Dask across Linux and Windows. As expected, a decreasing tendency of the import/merge times with the increase of the working nodes can be seen. When testing Spark on the import and merge benchmark, both Windows and Linux ran out of memory with two and four workers. Swap memory could be used to overcome this shortcoming, however, the resulting comparison would not be fair because the Dask benchmarks didn't need the swap memory.

[1]Scripts for the benchmarks, https://github.com/sensorlab/parMLBenchmarks

[2]EMINST dataset - https://www.kaggle.com/crawford/emnist (accessed: 30.07.2022)

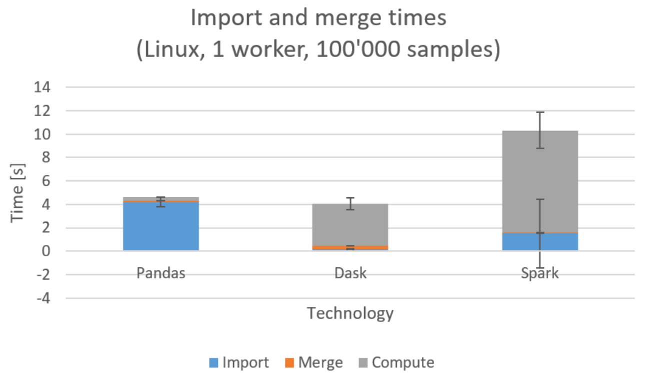


Fig. 2. Benchmark results of import and merge times at 100k samples: raw data to Pandas.

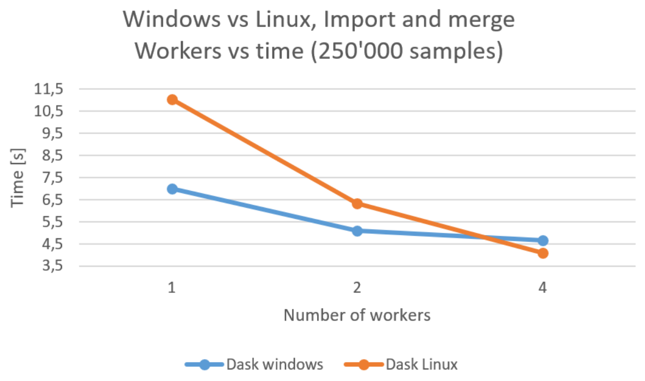


Fig. 3. Benchmark results two operating systems, Dask with import and merge on 250k samples.

B. Machine learning

Figure 4 shows the comparison of computation time between Dask, Spark, and SciKit on the Windows operating system for different dataset sizes. Each column in the figure represents the average computation time of 5 test runs. The results show that Dask and Spark are almost equivalent when the input dataset size is around 150k samples. Dask performs better on smaller datasets, while Spark's performance is best on larger datasets. Interestingly, SciKit outperforms both Dask and Spark on all dataset sizes, although it is not able to parallelize tasks. This is most likely because of the transfer times between nodes and the overall overhead of Dask and Spark. Since the datasets fit completely into the computer's memory, SciKit has no problems computing them, while Dask and Spark only cause unnecessary overhead. However, Dask and Spark are meant for large clusters with hundreds or even thousands of nodes, while SciKit is meant for computations on a single computer.

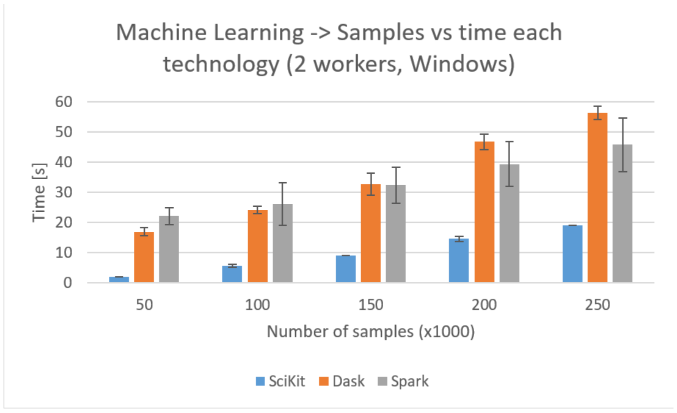


Fig. 4. Computational time for different dataset sizes on Windows operating system.

Figure 5 shows the results of the same experiment performed on the Linux operating system. Compared to the Figure 4, the results are very similar, with only difference that on Linux operating system Dask performs better then Spark even when input data set contains 150k samples.

Table I shows the F1 scores. An F1 score is the harmonic mean (alternative metric for the arithmetic mean) of precision and recall. The precision gives information on how many of the predicted samples that have been predicted as positive are correct. The recall gives information on how many of all positive samples the model managed to find.

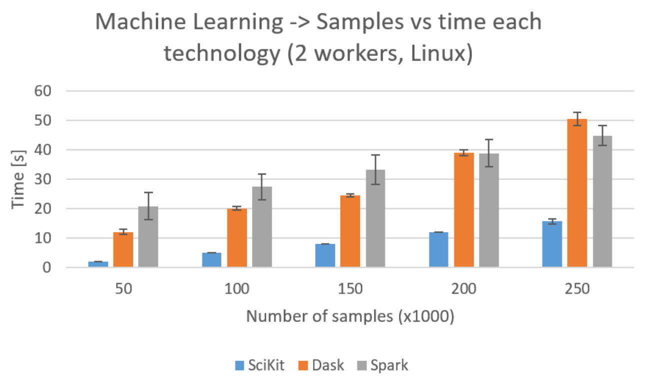


Fig. 5. Computational time for different dataset sizes on Linux operating system.

TABLE I
TABLE OF F1 SCORES FOR WINDOWS BENCHMARKS FOR VARIOUS SAMPLE SIZES (SIMILAR FOR LINUX).

	Number of samples (x1000)				
	50	100	150	200	250
Spark	0.71	0.73	0.73	0.71	0.71
Dask	0.71	0.72	0.73	0.71	0.70
Scikit	0.70	0.71	0.70	0.71	0.73

The machine learning benchmark measured the time to cast all columns into smaller data types. It seems that Dask has a dedicated function to cast all of the columns of a Dask dataframe at once whereas with the Spark function you have to cast each column one by one. The Dask casting was faster (0.06s) than Sparks (7.2s).

V. CONCLUSIONS

In this paper we benchmarked two parallel computing technologies, Dask and Apache Spark, against each other and against the single node SciKit Learn. The benchmarks were computed on the EMNIST dataset for various subsets from 50k to 250k samples on different operating systems and various degrees of parallelization. The results show a slight advantage on running the training pipeline on Linux rather than on Windows. Dask is seen as superior in dataframe manipulation while Apache Spark has a superior end-to-end processing performance on larger datasets with comparable final F1 scores.

ACKNOWLEDGMENTS

This work was funded in part by the Slovenian Research Agency under the grant P2-0016.

REFERENCES

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.
[2] M. Rocklin, "Dask: Parallel computation with blocked algorithms and task scheduling," in *Proceedings of the 14th python in science conference*, vol. 130, p. 136, Citeseer, 2015.
[3] S. Celis and D. R. Musicant, "Weka-parallel: machine learning in parallel," in *Carleton College, CS TR*, Citeseer, 2002.
[4] M. Dugré, V. Hayot-Sasson, and T. Glatard, "A performance comparison of dask and apache spark for data-intensive neuroimaging pipelines," in *2019 IEEE/ACM Workflows in Support of Large-Scale Science (WORKS)*, pp. 40–49, 2019.
[5] P. Mehta, S. Dorkenwald, D. Zhao, T. Kaftan, A. Cheung, M. Balazinska, A. Rokem, A. Connolly, J. Vanderplas, and Y. AlSayyad, "Comparative evaluation of big-data systems on scientific image analytics workloads," vol. 10, p. 1226–1237, VLDB Endowment, aug 2017.
[6] M. H. Nguyen, J. Li, D. Crawl, J. Block, and I. Altintas, "Scaling deep learning-based analysis of high-resolution satellite imagery with distributed processing," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 5437–5443, 2019.
[7] M. T. S. J. William Cheng, Ioannis Paraskevakos, "Image processing using task parallel and data parallel frameworks," pp. 1–7, 2019.
[8] S. Chintapalli, D. Dagit, B. Evans, R. Farivar, T. Graves, M. Holderbaugh, Z. Liu, K. Nusbaum, K. Patil, and B. J. Peng, "Benchmarking streaming computation engines: Storm, flink and spark streaming," 2016.

An Efficient Implementation of Hubness-Aware Weighting Using Cython

Krisztian Buza
buza@biointelligence.hu
BioIntelligence Group, Department of Mathematics-Informatics
Sapientia Hungarian University of Transylvania
Targu Mures, Romania

ABSTRACT

Hubness-aware classifiers are recent variants of k-nearest neighbor. When training hubness-aware classifiers, the computationally most expensive step is the calculation of hubness scores. We show that this step can be sped up by an order of magnitude or even more if it is implemented in Cython instead of Python while the accuracy is the same in both cases.

KEYWORDS

nearest neighbor, hubs, cython

1 INTRODUCTION

Nearest neighbor classifiers are simple, intuitive and popular, there are theoretical results about their accuracy and error bounds [6]. However, nearest neighbors are affected by *bad hubs*. An instance is called a bad hub, if it appears surprisingly frequently as nearest neighbor of other instances, but its class label is different from the labels of those other instances. Bad hubs were shown to be responsible for a surprisingly large fraction of the total classification error [10].

In order to reduce the detrimental effect of bad hubs, hubness-aware classifiers have been introduced, such as Hubness-Weighted k-Nearest Neighbor (HWKNN) [9], Naive Hubness Bayesian Nearest Neighbor (NHBNN) [16] and Hubness-based Fuzzy Nearest Neighbor (HFNN) [14]. Hubness has also been studied in context of collaborative filtering [8], regression [3], clustering [15], instance selection and feature selection [13]. Recently, hubness-aware ensembles have been proposed [17] and used for the classification of breast cancer subtypes [12].

Other prominent applications of hubness-aware methods include music recommendation [7], time series classification [11], drug-target prediction [4] and classification of gene expression data [2]. Last, but not least, we mention that even neural networks may benefit from hubness-aware weighting [5].

Hubness-aware classifiers may be implemented in various programming languages, one of the most prominent implementation is probably the Java-based HubMiner[1] library.

[1]https://github.com/datapoet/hubminer

In case of the aforementioned hubness-aware classifiers, the computationally most expensive step of the training is to determine the *hubness scores* of training instances, i.e., how frequently they appear as (bad) nearest neighbors of other instances. In this paper, we address this issue by a Cython-based implementation. Cython [1] aims to combine the advantages of Python (rapid prototyping and clarity thanks to concise code) with the efficiency of C. In particular, we implement the computation of hubness scores in Cython. Compared with a standard implementation in Python, we observed up to 25 times speedup on the Spambase dataset[2] from the UCI repository (and the speedup is likely to be even more in case of larger datasets).

2 BACKGROUND: HUBNESS-AWARE WEIGHTING

We say that an instance x is a *bad neighbor* of another instance x' if (i) x is one of the k-nearest neighbors of x' and (ii) their class labels are different. In case of hubness-aware weighting [9], first we determine how frequently each instance x appears as bad neighbor of other instances. This is denoted as $BN_k(x)$. Subsequently, the normalized bad hubness score $h_b(x)$ of each instance x is calculated as follows:

$$h_b(x) = \frac{BN_k(x) - \mu(BN_k)}{\sigma(BN_k)} \tag{1}$$

where $\mu(BN_k)$ and $\sigma(BN_k)$ denote the mean and standard deviation of the $BN_k(x)$ values over all instances of the training data. HWKNN performs weighted k-nearest neighbor classification, the weight of each training instance is $w(x) = e^{-h_b(x)}$. For a detailed illustration of HWKNN we refer to [13].

3 CYTHON-BASED IMPLEMENTATION OF HUBNESS CALCULATIONS

Python code is usually run by an interpreter which makes the execution relatively slow. Much of the inefficiency originates from dynamic typing: for example, the actual semantic of the '+' symbol depends on the types of the operands. It may stand for addition of numbers, concatenation of strings or lists, element-wise addition of arrays, etc. Which of the operations to perform, will be determined by the interpreter at execution time.

The core idea of Cython[3] is to annotate variables according to their types and to compile the resulting code into C which will further be compiled into binary code for efficient execution. In case of computationally expensive functions, this may results in

[2]https://archive.ics.uci.edu/ml/datasets/spambase
[3]https://cython.org/

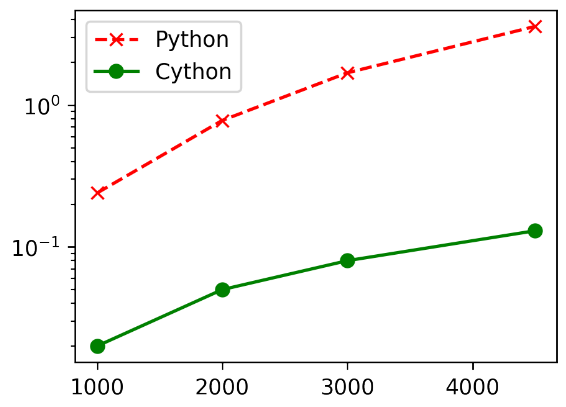


Figure 1: Runtime (in second, vertical axis) of hubness score calculation in case of Python-based (dashed line with 'x') and Cython-based (solid line with bullets) implementations for various number of instances (horizontal axis).

several orders of magnitude speedup. At the same time, functions implemented in Cython can be called from Python code just like Python functions.

We implemented the calculation of hubness scores both in Python and Cython, and made the code available in our github repository:

https://github.com/kr7/cython .

We evaluated both implementations on the Spambase dataset from the UCI repository. The dataset contains 4601 instances and 57 features (without the class label). Each instance corresponds to an e-mail. For each e-mail, the same features were extracted. The associated classification task is to decide whether the e-mail is spam or not.

We used 100 instances as test data and 4500 instances as training data. We run the experiments in Google Colab.[4] We used $k = 10$ nearest neighbors both for the calculation of hubness scores and the final classification. According to our observations, the Cython-based calculation of hubness scores was more than 20 times faster than the standard implementation in Python. Both versions produced the exactly same $BN_k(x)$ scores. As the weight of an instance x only depends on its $BN_k(x)$ score, both versions produce the same predictions. Therefore the accuracy (0.94) is equal in both cases.

We repeated the experiments with using only 1000, 2000 and 3000 instances as training data. As Fig. 1 shows, the Cython-based implementation was consistently faster than the implementation in Python. Note that logarithmic scale is used on the vertical axis. The difference showed an increasing trend when more training data was used: whereas in case of 1000 training instances, the Cython-based implementation was only about 12 times faster than the Python-based implementation, in case of 4500 training instances, the speedup factor was approximately 25. This may be attributed to the non-linear complexity of hubness score calculations. Assuming a naive implementation, determination of the nearest neighbors of an instance is linear in the size of the training data. However, in order to calculate the hubness scores, the nearest neighbors of *all* the training instances have to be determined. Thus the resulting overall complexity is quadratic.

We note that, both in case of Cython and Python, indexing techniques may be used to speed up the determination of the nearest neighbors. However, we omitted indexing in our implementation for simplicity.

[4]https://colab.research.google.com

4 DISCUSSION

In order to calculate distances effectively, we used pairwise distances from scikit-learn in our experiment. However, in case of *large* datasets, it may be necessary to calculate distances on the fly, as the distance matrix may be too large to be stored in RAM. In such cases, it may be worth considering to implement the distance calculations in Cython as well. In our previous works, we observed that the calculation of dynamic time warping distance was several orders of magnitudes faster when we implemented it in Cython instead of Python.

In case of *very large* datasets, straight forward calculation of hubness scores may be infeasible due to its quadratic complexity even if the calculations are implemented in Cython. In such cases, the aforementioned indexing techniques and/or calculation of approximate hubness scores (e.g. using a random subset of the data) may be necessary.

As future work, we plan an exhaustive evaluation of both implementations with respect to various datasets with different sizes and number of features.

ACKNOWLEDGEMENT

The author thanks to the Reviewers for their insightful comments and suggestions.

REFERENCES

[1] Stefan Behnel, Robert Bradshaw, Craig Citro, Lisandro Dalcin, Dag Sverre Seljebotn, and Kurt Smith. 2010. Cython: The best of both worlds. *Computing in Science & Engineering* 13, 2 (2010), 31–39.
[2] Krisztian Buza. 2016. Classification of gene expression data: A hubness-aware semi-supervised approach. *Computer methods and programs in biomedicine* 127 (2016), 105–113.
[3] Krisztian Buza, Alexandros Nanopoulos, and Gábor Nagy. 2015. Nearest neighbor regression in the presence of bad hubs. *Knowledge-Based Systems* 86 (2015), 250–260.
[4] Krisztian Buza and Ladislav Peška. 2017. Drug–target interaction prediction with Bipartite Local Models and hubness-aware regression. *Neurocomputing* 260 (2017), 284–293.
[5] Krisztian Buza and Noémi Ágnes Varga. 2016. Parkinsonet: estimation of updrs score using hubness-aware feedforward neural networks. *Applied Artificial Intelligence* 30, 6 (2016), 541–555.
[6] Luc Devroye, László Györfi, and Gábor Lugosi. 2013. *A probabilistic theory of pattern recognition.* Vol. 31. Springer Science & Business Media.
[7] Arthur Flexer, Monika Dörfler, Jan Schlüter, and Thomas Grill. 2018. Hubness as a case of technical algorithmic bias in music recommendation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW).* IEEE, 1062–1069.
[8] Peter Knees, Dominik Schnitzer, and Arthur Flexer. 2014. Improving neighborhood-based collaborative filtering by reducing hubness. In *Proceedings of International Conference on Multimedia Retrieval.* 161–168.
[9] Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2009. Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In *Proceedings of the 26th Annual International Conference on Machine Learning.* 865–872.
[10] Milos Radovanovic, Alexandros Nanopoulos, and Mirjana Ivanovic. 2010. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research* 11, sept (2010), 2487–2531.
[11] Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. 2010. Time-series classification in many intrinsic dimensions. In *Proceedings of the 2010 SIAM International Conference on Data Mining.* SIAM, 677–688.

[12] S Raja Sree and A Kunthavai. 2022. Hubness weighted svm ensemble for prediction of breast cancer subtypes. *Technology and Health Care* 30, 3 (2022), 565–578.
[13] Nenad Tomašev, Krisztian Buza, Kristóf Marussy, and Piroska B Kis. 2015. Hubness-aware classification, instance selection and feature construction: Survey and extensions to time-series. In *Feature selection for data and pattern recognition*. Springer, 231–262.
[14] Nenad Tomašev, Miloš Radovanovic, Dunja Mladenic, and Mirjana Ivanovic. 2011. A probabilistic approach to nearest-neighbor classification: Naive hubness bayesian knn. In *Proc. 20th ACM Int. Conf. on Information and Knowledge Management (CIKM)*. 2173–2176.
[15] Nenad Tomasev, Milos Radovanovic, Dunja Mladenic, and Mirjana Ivanovic. 2013. The role of hubness in clustering high-dimensional data. *IEEE transactions on knowledge and data engineering* 26, 3 (2013), 739–751.
[16] Nenad Tomašev, Miloš Radovanović, Dunja Mladenić, and Mirjana Ivanović. 2014. Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. *International Journal of Machine Learning and Cybernetics* 5, 3 (2014), 445–458.
[17] Qin Wu, Yaping Lin, Tuanfei Zhu, and Yue Zhang. 2020. HIBoost: A hubness-aware ensemble learning algorithm for high-dimensional imbalanced data classification. *Journal of Intelligent & Fuzzy Systems* 39, 1 (2020), 133–144.

Semantic Similarity of Parliamentary Speech using BERT Language Models & fastText Word Embeddings

Katja Meden
Department of Knowledge Technologies E8,
Jožef Stefan Institute
katja.meden@ijs.si

ABSTRACT

The main objective of this paper is to present the work done on comparing the two methods for measuring semantic similarity of parliamentary speech between coalition and opposition regarding the adoption of the first COVID-19 epidemic response package. We first measured sentence similarity using four BERT-based language models (Language agnostic BERT Sentence Encoder - LaBSE model, Sentence-LaBSE, Sentence-BERT, multilingual BERT - mBERT) and compared the results amongst them. Using the word embedding method, fastText, we then measured the semantic similarity of full-text parliamentary speech and presented the results using descriptive analysis. Lastly, we compared the usage of both methods and highlighted some of the advantages and disadvantages of each method for measuring the semantic similarity of parliamentary speech.

KEYWORDS

parliamentary speech, semantic similarity, sentence similarity, BERT language models, fastText

1 INTRODUCTION

"National parliamentary data is a verified communication channel between the elected political representatives and society members in any democracy. It needs to be made accessible and comprehensive - especially in times of a global crisis." [13] In parliamentary discourse, politicians expound their beliefs and ideas through argumentation and to persuade the audience, they highlight some aspect of an issue. If we are to understand the role of parliamentary discourse practices, we need to explore the recurring linguistic patterns and rhetorical strategies used by MPs that help to reveal their ideological commitments, hidden agendas, and argumentation tactics [11]. One of the ways to study the aforementioned linguistic patterns can be done by researching similarities of parliamentary speeches using different methods for measuring semantic similarity of text.

The aim of this paper is to present the work done on comparing the two methods for measuring semantic similarity of parliamentary speech between coalition and opposition regarding the adoption of the first COVID-19 epidemic response package.

We measured sentence similarity with four BERT-based language models (Language agnostic BERT Sentence Encoder - LaBSE model [7], Sentence-LaBSE [8], Sentence-BERT [14], multilingual BERT – mBERT [1]) and compared the scores of most similar and least similar sentences.

To facilitate the intended scope of our initial research, i.e., researching similarity of full-text parliamentary speech, we used fastText [5] and presented results using descriptive analysis to gain additional insight into the characteristics of coalition and opposition parliamentary speech. Lastly, we highlighted some of the advantages and disadvantages of each method for measuring semantic similarity of parliamentary speech.

The paper is structured as follows: Section 2 contains an overview of the related work on word embeddings and language models. Section 3 presents the methodology and we describe the experiment setting in Section 4. The experiment results are found in Section 5. Finally, we conclude the paper and provide ideas for future work in Section 6.

2 RELATED WORK

Two blocks of texts are considered similar if they contain the same words or characters. Techniques like Bag of Words (BoW), Term Frequency - Inverse Document Frequency (TF-IDF) can be used to represent text as real value vectors to aid calculation of Semantic Textual Similarity (STS) [3]. STS is defined as the measure of semantic equivalence between two blocks of text and usually give a ranking or percentage of similarity between texts, rather than a binary decision as similar or not similar [3]. Word embeddings are one of the methods developed to aid in measuring semantic similarity. They provide vector representations of words where vectors retain the underlying linguistic relationship between the similarities of the words. Word embeddings consist of two types: static and contextualized word embeddings. With static word embeddings, words will always have the same representation, regardless of the context where it occurs, while with contextualized word embedding, representation depends on the context of where that word occurs – meaning, that the same word in different contexts can have different representations.

FastText is an open-source, free, lightweight library that allows users to learn text representations and text classifiers [5]. It is a representative of the static word embedding technique, where a vector representation is associated to each character n-gram; words being represented as the sum of these representations [2]. The fundamental problem of word embeddings is that they generate the same embedding for the same word in different contexts, failing to capture polysemy [4].

Language models are contextualized word representations that aim at capturing word semantics in different contexts to address the issue of polysemy and the context of words [4]. BERT, or Bidirectional Encoder Representations from Transformer, is a language model, designed to pre-train deep bidirectional representations from an unlabeled text by jointly conditioning on both left and right context in all layers [6]. BERT word representations are therefore contextual

3 METHODOLOGY

3.1 Dataset

Dataset contains 230 documents (speeches) from the Extraordinary Session 33 from the corpora of the Slovenian parliamentary debates (ParlaMint-SI) [9] from 2014 to mid-2020, linguistically annotated and represented in the CoNNL-U format (which include POS, lemmatized and NER tags**)**. We chose an extraordinary session in a time of crisis for two reasons: firstly, regular sessions deal with multiple problems (such as MP questions), which makes a comparison between speeches difficult. Similarly, we chose only one specific theme (the adoption of the first epidemic response package), which helped in the initial analysis and comparison of documents.

3.2 Data analysis and pre-processing

For the initial data analysis, we used the Orange data-mining tool [12] that helped us with the data understanding and initial dataset pre-processing.

For full speech measuring with fastText we removed speeches by Chairperson to avoid adding noise to the dataset in the form of procedural speech that would make measuring semantic similarities almost impossible. We also removed Slovene stopwords and manually added a list of four additional stopwords: *hvala*, *danes*, l*epa* and *beseda*, which excluded the very common phrase *Hvala za besedo* (eng. Thank you for the word) and its variations. Some of the documents were missing the *party_status* labels (values: *coalition* and *opposition*). The missing values (17 documents) were thus removed from the dataset. The pre-process gave us a total of 97 documents, presented in Table 1. Looking at the distributions of the speeches in the session, almost 1/3 of the speeches belongs to the opposition. Both coalition and opposition consists of four political parties: LMŠ, Levica, SAB and SD are part of the opposition, all of mostly left and centre-left political orientation. Similarly, the coalition consists of DeSUS, NSi, SDS and SMC political parties[1], all mostly right-winged and centre-right parties.

Table 1: Preprocessed dataset

Sample	Number of documents	Total
Coalition	30 (30.93%)	97
Opposition	67 (69.07%)	

[1] Technically, the opposition consists of 5 political parties, but SNS (Slovenska Nacionalna Stranka) does not have any speeches in the dataset.

We used the same settings for the second part of the experiment (comparing sentence similarity with the four BERT-based models) with one difference. Since all BERT-based models support *max_lenght* input in the size 512 tokens, we decided to filter out sentences that refer explicitly to the response package (keyword for selection being *zakon*). To facilitate the visualisations and balance out our dataset, we randomly chose 20 sentences for each group (coalition/opposition).

3.3 Experiment settings

As mentioned, BERT-based models have restrictions on the maximum length of input documents. For most, this is 512 tokens, and in the case of Sentence-BERT, this restriction is even more severe (128 word tokens). Most speeches in the dataset are longer than the maximum length – this limitation did not allow us to conclude semantic similarity measurement on full parliamentary speech. The first part of the experiment therefore focuses on sentence similarity. From previously described BERT-based models, three of the models were fine-tuned for sentence similarity tasks: Sentence-LaBSE [7], LaBSE [8], mBERT [1] and Sentence-BERT [14]. For easier comparison, we used mean pooling and cosine distance to measure the similarity.

To achieve the intended scope of our initial research (researching the semantic similarity of parliamentary speech), we used the fastText-based Orange widget *Document embedding* (using mean as the aggregation method) to embed our documents and calculate cosine similarity to achieve comparison between coalition and opposition parliamentary speech. With these two experiments, we can compare measuring semantic similarity with language models to the word embedding method (fastText). This comparison would be better with Longformer language model (which can take up to around 1000+ word tokens as *max_input*) as we could compare methods for measuring semantic similarity of full-text documents (speeches), but as of time of writing this paper, Longformer [10] does not yet support Slovene language.

4 RESULTS

4.1 Results of the sentence similarity measure with BERT-based models

As stated previously, we used four different BERT-based models to measure semantic similarity of 40 sentences (20 sentences for each group - coalition and opposition) and visualized the results using heat maps (example in Figure 1). Initially, we first selected well-known BERT-based models that were optimized for Slovene (trilingual model CroSloEngual BERT and monolingual model SloBERTa), that did not produce reliable results - as shown in Table 2, CroSloEngual [15] and SloBERTa [16] produce extremely high similarity scores, since, as we later discovered, were not fine-tuned for sentence similarity task.

Table 2: Similarity scores of language models for most similar and least similar sentences

Model	Most similar	Least similar
Sentence-LaBSE	0.6184	0.1235
LaBSE	0.7610	0.3649
mBERT	0.8930	0.5377
Sentence-BERT	0.6677	-0.0792
CroSloEngual	0.9931	0.9480
SloBERTa	0.9867	0.8899

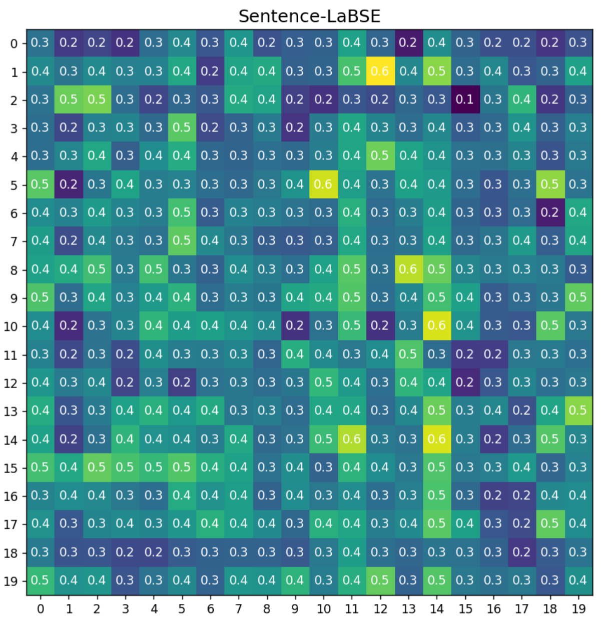


Figure 1: Example of heat map using Sentence-LaBSE model

When comparing the models, it does not surprise that Sentence-LaBSE and Sentence-BERT show very similar results (see Table 2), as they come from the same family of models and thus have similar model architecture (and are both fine-tuned for this specific task). What is interesting is the fact that Sentence-BERT is the only model that produced a negative score for the least similar sentence (similarity score of -0.0792), while mBERT model showed the highest similarity scores (outside of CroSloEngual and SloBERTa). Some of the highest scored sentences showed that speakers from different party statuses tend to use similar language patterns, for example:

Coalition*: "Ob hitrem sprejemanju zakona je potrebno zagotoviti, da ne bodo spregledane posamezne ranljive skupine posameznikov."*
(Eng. "With the rapid adoption of the law, it is necessary to ensure that individual vulnerable groups of individuals are not overlooked.")

Opposition*: "Še enkrat, ostaja še cela vrsta ranljivih skupin v zakonu, ki je nenaslovljena."*
(Eng. "Once again, there is a whole range of vulnerable groups in the law that remain unaddressed.")

4.2 Results of the document similarity with fastText

For the second part of our experiment, we used fastText for word embedding and measured cosine distance to get semantic similarity score of our documents. Figure 2 shows visualized results comparing speeches between coalition and opposition speakers:

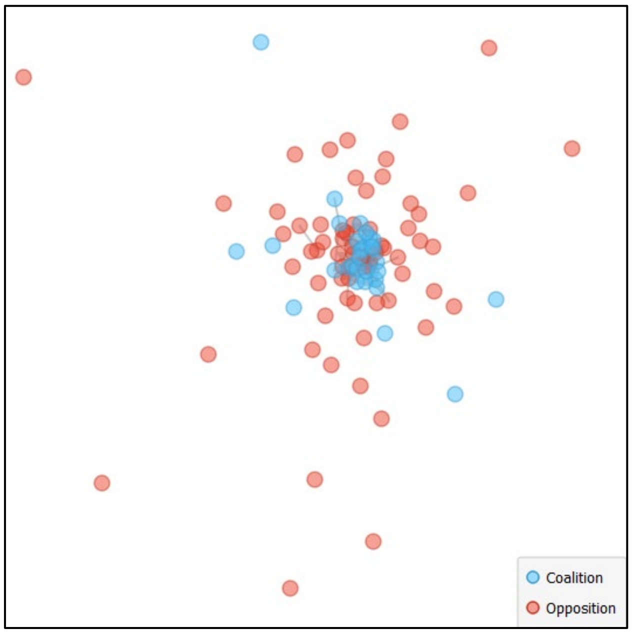


Figure 2: Document similarity with fastText, visualized using MDS

Documents (or speeches) are connected closely together – this could be attributed mostly to the fact that they are addressing the same issue – the adoption of the first epidemic release package. The most similar speeches were made by the members of the political party SDS (coalition) and SD (opposition), followed closely by SMC and Levica. All speeches are long and focus on the topic of the session – the proposed law (most speeches include keywords such as "*zakon*" (law), "*zakonski paket*" (law packet), "*amandma*" (amendment), "*ukrepi*" (measures).

Outlier detection analysis showed 8 speeches (7 made by the opposition, 1 by coalition), which are all very short and focus solely on parliamentary procedures. We also observed some trends in the usage of the words, concatenated from the word "*korona*": "*koronakriza*", "*koronazakon*", "*antikoronazakon*", "*koronaobveznica*", "*koronapomoči*", "*protikoronapaket*" etc. (used mostly by the opposition).

In Figure 3, we compared speech between the members of the opposition. The visualization showed a cluster of similar speeches. Members of Levica seemed to be most vocal during the session (by having more than 50% of all opposition speeches), while also having several similar speeches, with the central sub-topic being proposed amendments to the law and financial consequences of it. The least similar speech was made by Violeta Tomić, member of Levica, in regard to the date the epidemic was declared.

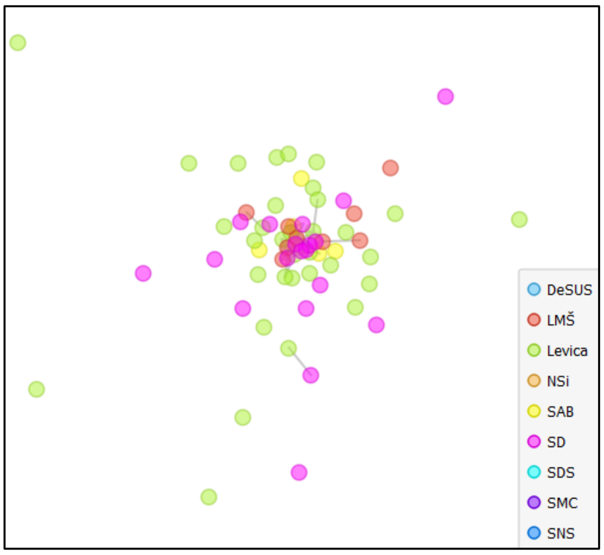


Figure 3: Document similarity with fastText (opposition)

In Figure 4, we compared speech between the members of coalition: speeches are less connected; with most similar divided among SDS members, closely connected to the SMC, NSI and DeSUS members. The common sub-topic to all of the speeches made is the financial crisis as a direct result of the epidemic. Two of the most far-away speeches belong to the member of DeSUS (Franc Jurša). Both speeches are among the shortest ones in the dataset, with a focus on the topic of pensions and registration of a parliamentary group, and thus are not explicitly connected to the central topic of the discourse.

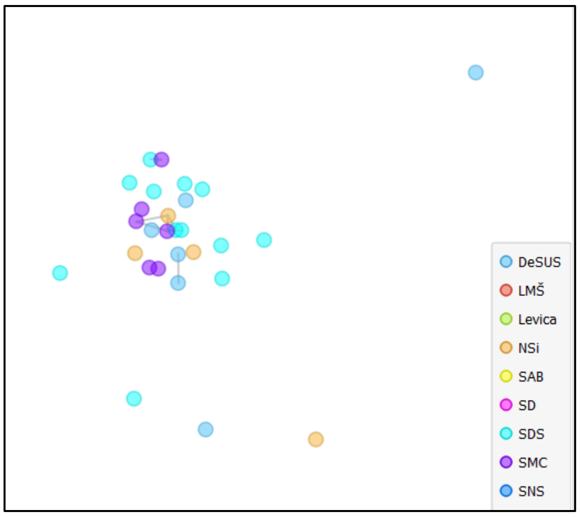


Figure 4: Document similarity with fastText (coalition)

5 CONCLUSIONS

In this paper, we were comparing language models and word embeddings as methods for measuring semantic similarity of parliamentary speech. In the initial stages, it turned out that there is not a lot of models that support Slovene as input language. Those that were made explicitly with Slovene in mind (such as SloBERTa and CroSloEngual BERT) were not fine-tuned for semantic similarity/sentence similarity tasks and thus do not produce accurate results. Limitation on maximum length of input text that most BERT-based models have is probably one of the biggest disadvantages of the language models for semantic similarity measures (this is being alleviated with new emerging language models, such as Longformer, that allow over 1000+ tokens as maximum input length). For sentence similarity task language models from Sentence-BERT family show the most accuracy and are easier to use as standard BERT models (such as mBERT).

Even though BERT contextualizes word embeddings (and therefore might produce better results because of it), fastText solved the problem of text-input length and combined with Orange data mining tool allowed us to explore similarities between speeches as we originally intended to do. From the document similarity analysis, we saw that most speeches were relatively connected (similar) to one another. Speeches amongst the members of the opposition were more similar in comparison to the speeches made amongst coalition members. There were a few outlier speeches in both opposition and coalition – they were all shorter speeches and less related to the original topic of the discourse. For future work, some limitations of this research should first be addressed (e.g., comparing language models to word embedding techniques on a full-text basis) and repeat the experiments with fine-tuned SloBERTa and CroSloEngual model on full ParlaMint-SI corpora.

REFERENCES

[1] BERT multilingual base model (cased): https://huggingface.co/bert-base-multilingual-cased

[2] Bojanowski, Piotr, Grave, Edouard, Joulin, Armand and Mikolov, Tomas. (2017). Enriching word vectors with subword information. In *Transactions of the Association for Computational Linguistics*, 5, 135-146. DOI: https://doi.org/10.1162/tacl_a_00051

[3] Chandrasekaran, Dhivya, and Vijay Mago. 2021. Evolution of Semantic Similarity—A Survey. In *ACM Computing Surveys,* 1-37.

[4] David S. Batista. 2018. Language Models and Contextualised Word Embeddings. https://www.davidsbatista.net/blog/2018/12/06/Word_Embeddings/

[5] FastText - Library for efficient text classification and representation learning. https://fasttext.cc/

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[7] Language-agnostic BERT Sentence Encoder (LaBSE) (Sentence-Transformers): https://huggingface.co/sentence-transformers/LaBSE

[8] Language-agnostic BERT Sentence Encoder (LaBSE): https://huggingface.co/setu4993/LaBSE

[9] Linguistically annotated multilingual comparable corpora of parliamentary debates ParlaMint.ana 2.1. 2021. http://hdl.handle.net/11356/1431

[10] Longformer: https://huggingface.co/docs/transformers/model_doc/longformer

[11] Naderi, Nona, and Graeme Hirst. 2015. Argumentation mining in parliamentary discourse. In Principles and practice of multi-agent systems, 16-25. https://cmna.csc.liv.ac.uk/CMNA15/paper%209.pdf

[12] Orange: Data Mining Tool for visual programming. https://orangedatamining.com/

[13] ParlaMint: Towards Comparable Parliamentary Corpora. 2020. https://www.clarin.eu/content/parlamint-towards-comparable-parliamentary-corpora

[14] Sentence-BERT (sentence-transformers/distiluse-base-multilingual-cased-v2): https://huggingface.co/sentence-transformers/distiluse-base-multilingual-cased-v2

[15] Ulčar, Matej and Robnik-Šikonja, Marko, 2020, *CroSloEngual BERT 1.1*, Slovenian language resource repository CLARIN.SI, http://hdl.handle.net/11356/1330.

[16] Ulčar, Matej and Robnik-Šikonja, Marko, 2021, *Slovenian RoBERTa contextual embeddings model: SloBERTa 2.0*, Slovenian language resource repository CLARIN.SI, http://hdl.handle.net/11356/1397

Indeks avtorjev / Author index

Odkrivanje znanja in podatkovna skladišča - SiKDD

Data Mining and Data Warehouses - SiKDD

Urednika • Editors:
Dunja Mladenić, Marko Grobelnik