Identification of COVID-19 Related Fake News via Neural Stacking

Boshko Koloski[1,2](✉), Timen Stepišnik-Perdih[3], Senja Pollak[1], and Blaž Škrlj[1,2]

[1] Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
{boshko.koloski,blaz.skrlj}@ijs.si
[2] Jožef Stefan Int. Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia
[3] University of Ljubljana, Faculty of Computer and Information Science, Večna pot 113, Ljubljana, Slovenia

Abstract. Identification of Fake News plays a prominent role in the ongoing pandemic, impacting multiple aspects of day-to-day life. In this work we present a solution to the shared task titled *COVID19 Fake News Detection in English*, scoring the 50th place amongst 168 submissions. The solution was within 1.5% of the best performing solution. The proposed solution employs a heterogeneous representation ensemble, adapted for the classification task via an additional neural classification head comprised of multiple hidden layers. The paper consists of detailed ablation studies further displaying the proposed method's behavior and possible implications. The solution is freely available.
https://gitlab.com/boshko.koloski/covid19-fake-news

Keywords: Fake-news detection · Stacking ensembles · Representation learning

1 Introduction

Fake news can have devastating impact on the society. In the times of a pandemic, each piece of information can have a significant role in the lives of everyone. The verification of the truthfulness of a given information as a fake or real is crucial, and can be to some extent learned [10]. Computers, in order to be able to solve this task, need the data represented in a numeric format in order to draw patterns and decisions. We propose a solution to this problem by employing various natural language processing and learning techniques.

The remainder of this work is structured as follows: Sect. 2 describes the prior work in the field of detection of fake-news. The provided data is described in Sect. 3 and Sect. 4 explains our proposed problem representation approaches while Sect. 5 introduces two different meta-models built on top of the basic representations listed in Sect. 4. The experiments and results achieved are listed in Sect. 6, finally the conclusion and the proposed future work are listed in Sect. 7.

T. Chakraborty et al. (Eds.): CONSTRAINT 2021, CCIS 1402, pp. 177–188, 2021.
https://doi.org/10.1007/978-3-030-73696-5_17

2 Related Work

The fake-news text classification task [16] is defined as follows: given a text and a set of possible classes *fake* and *real*, to which a text can belong, an algorithm is asked to predict the correct class of the text. Most frequently, fake-news text classification refers to classification of data from social media. The early proposed solutions to this problem used hand crafted features of the authors such as word and character feature distributions. Interactions between fake and real news spread on social media gave the problem of fake-news detection a network-alike nature [18]. The network based modeling discovered useful components of the fake-news spreading mechanism and led to the idea of the detection of bot accounts [17].

Most of the current state-of-the-art approaches for text classification leverage large pre-trained models like the one Devlin et al. [1] and have promising results for detection of fake news [4]. However for fake-news identification tasks, approaches that make use of n-grams and the Latent Semantic Analysis [2] proved to provide successful solutions on this task (see Koloski et al. [5]). Further enrichment of text representations with taxonomies and knowledge graphs [19] promises improvements in performance.

3 Data Description

In this paper we present a solution to the subset of the fake-news detection problem - The identification of COVID-19 related Fake News [10,11]. The dataset consists of social media posts in English collected from Facebook, Twitter and Instagram, and the task is to determine for a given post if it was real or fake in relation to COVID-19. The provided dataset is split in three parts: train, validation and test data. The distribution of data in each of the data sets is shown in Table 1.

Table 1. Distribution of the labels in all of the data splits.

Part	Train	Validation	Test
Size	6420	2140	2140
Real	3360 (52%)	1120 (52%)	1120 (52%)
Fake	3060 (48%)	1020 (48%)	1020 (48%)

4 Proposed Method

The proposed method consists of multiple submethods that aim to tackle different aspects of the problem. On one side we focus on learning the hand crafted features of authors and on the other we focus on learning the representation of the problem space with different methods.

4.1 Hand Crafted Features

Word Based. Maximum and minimum word length in a tweet, average word length, standard deviation of the word length in tweet. Additionally we counted the number of words beginning with upper and the number of words beginning a lower case.

Char Based. The character based features consisted of the counts of digits, letters, spaces, punctuation, hashtags and each vowel, respectively.

4.2 Latent Semantic Analysis

Similarly to Koloski et al. [5] solution to the PAN2020-Fake News profiling we applied the low dimensional space estimation technique. First we preprocessed the data by lower-casing the tweet content and removing the hashtags and punctuation. After that we removed the stopwords and obtained the final clean presentation. From the cleaned text, we generated the POS-tags using the NLTK library [6].

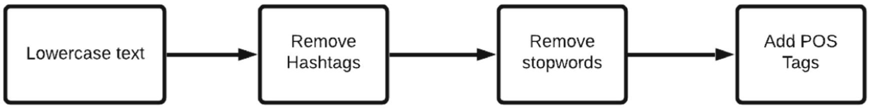

Fig. 1. Text preparation for the LSA.

For the feature construction space we used the technique used by Martinc et al. [8] which iterativly weights and chooses the best n-grams. We used two types of n-grams:

- Word based: n-grams of size 1 and 2
- Character based: n-grams of size 1, 2 and 3

We generated n features with $n/2$ of them being word and $n/2$ character n-grams. We calculated TF-IDF on them and preformed SVD [3] With the last step we obtained the LSA representation of the tweets.

For choosing the optimal number of features $\mathbf{n}$ and number of dimensions $\mathbf{d}$, we created custom grid consisted of $n' \in [500, 1250, 2500, 5000, 10000, 15000, 20000]$ and $d' \in [64, 128, 256, 512, 768]$. For each tuple $(n', d'), n' \in \mathbf{d}$ and $d' \in \mathbf{d}$ we generated a representation and trained (SciKit library [12]) SVM and a LR (Logistic regression) classifier. The learning procedure is shown in Fig. 2.

The best performing model had 2500 features reduced to 512 dimensions.

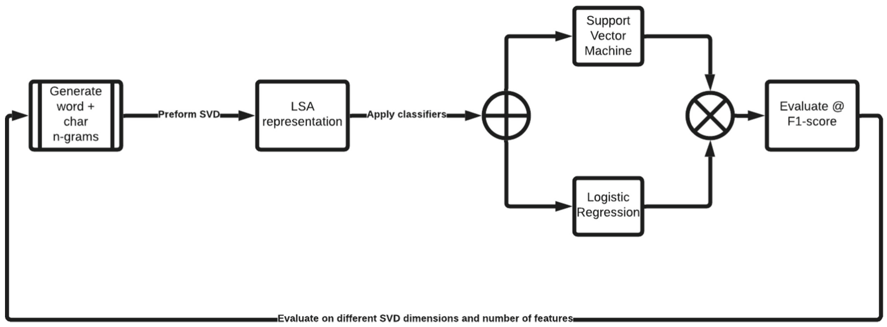

Fig. 2. The proposed learning procedure with the LSA. The evaluation is performed on validation dataset.

4.3 Contextual Features

We explored two different contextual feature embedding methods that rely on the transformer architecture. The first method uses the already pretrained *sentence_transfomers* and embedds the texts in an unsupervised manner. The second method uses DistilBERT which we fine tune to our specific task.

sentence_transfomers. For fast document embedding we used three different contextual embedding methods from the *sentence_transfomers* library [14]:

- *distilbert-base-nli-mean-tokens*
- *roberta-large-nli-stsb-mean-tokens*
- *xlm-r-large-en-ko-nli-ststb*

First, we applied the same preprocessing as shown in Fig. 1, where we only excluded the POS tagging step. After we obtained the preprocessed texts we embedded every tweet with a given model and obtained the vector representation. After we obtained each representation, we learned a Stochastic Gradient Descent based learner, penalizing both the "linear" and "hinge" loss parameters. The parameters were optimized on a GridSearch with a 10-fold Cross-validation on every tuple of parameters.

DistilBERT is a distilled version of BERT that retains best practices for training BERT models [15]. It is trained on a concatenation of English Wikipedia and Toronto Book Corpus. To produce even better results, we fine-tuned the model on train data provided by the organizers. BERT has its own text tokenizer and is not compatible with other tokenizers so that is what we used to prepare data for training and classification.

4.4 tax2vec Features

tax2vec [19] is a data enrichment approach that constructs semantic features useful for learning. It leverages background knowledge in the form of taxonomy or knowledge graph and incorporates it into textual data. We added generated semantic features using one of the two approaches described below to top 10000 word features according to the TF-IDF measure. We then trained a number of classifiers on this set of enriched features (Gradient boosting, Random forest, Logistic regression and Stochastic gradient descent) and chose the best one according to the F1-score calculated on the validation set. **Taxonomy based (tax2vec).** Words from documents are mapped to terms of the WordNet taxonomy [13], creating a document-specific taxonomy after which a term-weighting scheme is used for feature construction. Next, a feature selection approach is used to reduce the number of features. **Knowledge Graph based (tax2vec(kg)).** Nouns in sentences are extracted with SpaCy and generalized using the Microsoft Concept Graph [9] by "is_a" concept. A feature selection approach is used to reduce the number of features.

5 Meta Models

From the base models listed in Sect. 4 we constructed two additional meta-models by combining the previously discussed models.

5.1 Neural Stacking

In this approach we learn a dense representation with 5-layer deep neural network. For the inputs we use the following representations:

- LSA representation with $N = 2500$ features reduced to $d = 256$ dimensions.
- Hand crafted features - $d = 16$ dimensions
- *distilbert-base-nli-mean-tokens* - $d = 768$ dimensions
- *roberta-large-nli-stsb-mean-tokens* - $d = 768$ dimensions
- *xlm-r-large-en-ko-nli-ststb* - $d = 768$ dimensions

This represents the final input X_{Nx2576} for the neural network. After concatenating the representations we normalized them. We constructed a custom grid consisted of learning_rate = $\lambda \in$ [0.0001, 0.005, 0.001, 0.005, 0.01, 0.05, 0.1], dropout = $p \in$ [0.1, 0.3, 0.5, 0.7], batch_size $\in$ [16, 32, 64, 128, 256], epochs $\in$ [10, 100, 1000]. In the best configuration we used the *SELU* activation function and dropout $p = 0.7$ and learning rate $\lambda = 0.001$. The loss function was *Cross-Entropy* optimized with the *StochasticGradientOptimizer*, trained on $epochs = 100$ and with $batch\_size = 32$.
Layers were composed as following:

- *input* layer - $d = 2576$ nodes
- $dense_1$ layer - $d = 896$ nodes, activation = $SELU$
- $dense_2$ layer - $d = 640$ nodes, activation = $SELU$
- $dense_3$ layer - $d = 512$ nodes, activation = $SELU$
- $dense_4$ layer - $d = 216$ nodes, activation = $SELU$
- $dense_5$ layer - $d = 2$ nodes, activation = *Sigmoid*

5.2 Linear Stacking

The second approach for meta-learning considered the use of the predictions via simpler models as the input space. We tried two separate methods:

Final Predictions. We considered the predictions from the *LSA*, *DistilBert*, *dbert*, *xlm*, *roberta*, *tax2vec* as the input. From the models' outputs we learned a Stochastic Gradient Optimizer on 10-fold CV. The learning configuration is shown in Fig. 3.

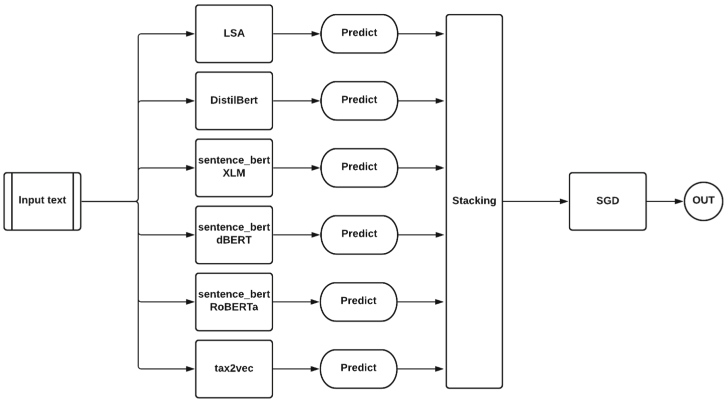

Fig. 3. Stacking architecture based on base model predictions.

Decision Function-Based Prediction. In this approach we took the given classifier's value of the decision function as the input in the stacking vector. For the SVM based SGD we used the *decision_function* and for the Logistic Regression we used the *Sidmoid_activation*. The proposed architecture is similar to the architecture in Fig. 3, where prediction values are replaced by decision function values.

6 Experiments and Results

This section describes model parameters, our experiments and the results of experiments as well as the results of the final submission.

We conducted the experiments in two phases. The experiment phases synced with the competition phases and were defined as TDT phase and CV phase. In the TDT phase the train and validation data is split into three subsets, while in the CV phase all data is concatenated and evaluated on 10-folds.

Table 2. Final chosen parameters for the best model of each vectorization.

Vectorization	Model	Parameters
LSA	LR	*'l1_ratio': 0.05, 'penalty': 'elasticnet', 'power_t': 0.5*
Hand crafted features	SVM	*'l1_ratio': 0.95, 'penalty': 'elasticnet', 'power_t': 0.1*
distilbert-base-nli-mean-tokens	LR	'C': 0.1, 'penalty': 'l2'
roberta-large-nli-stsb-mean-tokens	LR	'C': '0.01', 'penalty': 'l2'
xlm-r-large-en-ko-nli-ststb	SVM	'C': 0.1, 'penalty': 'l2'
Linear stacking_probs	SGD	'l1_ratio': 0.8, 'loss': 'hinge', 'penalty': 'elasticnet'
Linear stacking	SGD	'l1_ratio': 0.3, 'loss': 'hinge', 'penalty': 'elasticnet'
tax2vec_tfidf	SGD	'alpha': 0.0001, 'l1_ratio': 0.15, 'loss': 'hinge', 'power_t': 0.5
tax2vec(kg)_tfidf	SVM	'C': 1.0, 'kernel': 'rbf'

6.1 Train-Development-Test (TDT) Split

In the first phase, we concatenated the train and the validation data and splitted it into three subsets: *train* (75%), *dev* (18.75%) and *test* (6.25%). On the *train* split we learned the classifier which we validated on the *dev* set with measurement of F1-score. Best performing model on the *dev* set was finally evaluated on the test set. Achieved performance is presented in Table 3 and the best performances are shown in Fig. 4.

Table 3. F1-scores for different methods of vectorization on the TDT data split.

Vectorization	Train F1-score	DEV F1-score	Test F1-score
distilBERT-tokenizer	**0.9933**	**0.9807**	**0.9708**
Neural stacking	0.9645	0.9377	0.9461
Linear stacking	0.9695	0.9445	0.9425
tax2vec	0.9895	0.9415	0.9407
Linear stacking_probs	0.9710	0.9380	0.9390
LSA	0.9658	0.9302	0.9281
roberta-large-nli-stsb-mean-tokens	0.9623	0.9184	0.9142
xlm-r-large-en-ko-nli-ststb	0.9376	0.9226	0.9124
distilbert-base-nli-mean-tokens	0.9365	0.9124	0.9113
tax2vec(kg)	0.8830	0.8842	0.8892
Hand crafted features	0.7861	0.7903	0.7805

DistilBERT comes out on top in F1-score evaluation on all data sets in TDT data split—to the extent that we feared overfitting on the train data—while handcrafting features did not prove to be successful. Taxonomy based tax2vec feature construction trails distilBERTs score but using a knowledge graph to generalize constructed features seemed to decrease performance significantly (tax2vec(kg)). Other methods scored well, giving us plenty of reasonably good approaches to consider for the CV phase.

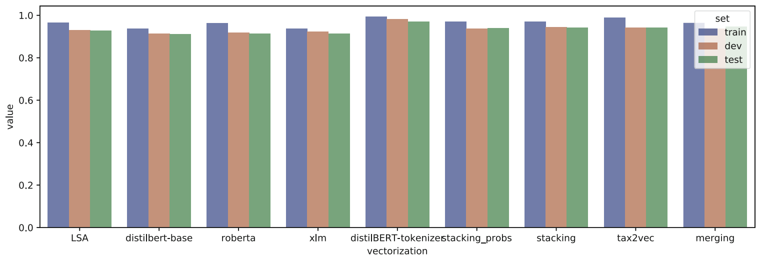

Fig. 4. Best performing methods of feature vectorization according to F1-score.

6.2 CV Split

In the second phase - the *CV* phase we concatenated the data provided by the organizers and trained models on 10-fold Cross-Validation. The evaluation of the best-performing models is presented in Table 4.

During cross-validation, LSA showed consistency in good performance. With similar performance were the tax2vec methods which this time scored very similarly.

Table 4. F1-scores of models when training using 10-fold cross-validation.

Model name	Vectorization	10-fold CV
LSA	LSA	**0.9436**
sentence_transformers	distilbert	0.9071
sentence_transformers	roberta-large	0.9077
sentence_transformers	xlm-roberta	0.9123
Gradient boosting	tax2vec	0.9335
Gradient boosting	tax2vec(kg)	0.9350

6.3 Evaluating Word Features

To better understand the dataset and trained models we evaluated word features with different metrics to pinpoint features with the highest contribution to classification or highest variance.

Features with the Highest Variance. We evaluated word features within the train dataset based on variance in *fake* and *real* classes and found the following features to have the highest variance:

"Fake" class – cure – coronavirus – video – president – covid – vaccine – trump – 19

"Real" class – number – total – new – tests – deaths – states – confirmed – cases – reported

SHAP Extracted Features. After training the models we also used Shapley Additive Explanations [7] to extract the most important word features for classification into each class. The following are results for the gradient boosting model:

"Fake" class – video – today – year – deployment – trump – hypertext transfer protocol

"Real" class – https – covid19 – invoking – laboratories – cases – coronavirus

Generalized Features. We then used WordNet with a generalizing approach called ReEx (Reasoning with Explanations)[1] to generalize the terms via the "is_a" relation into the following terms:

"Fake" class – visual communication – act – matter – relation – measure – hypertext transfer protocol – attribute

"Real" class – physical entity – message – raise – psychological feature

6.4 Results

Results of the final submissions are shown in Table 5.

Table 5. Final submissions F1-score results.

Submission name	Model	F1-score
btb_e8_4	Neural stacking	**0.9720**
btb_e8_3	LSA	0.9416
btb_e8_1	tax2vec	0.9219
btb_e8_2	Linear stacking	0.8464
btb_e8_5	distilbert	0.5059

[1] https://github.com/OpaqueRelease/ReEx.

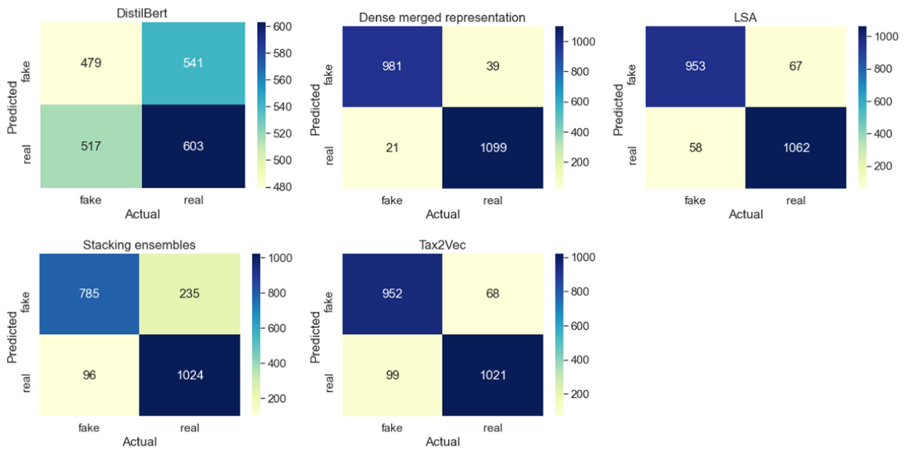

Fig. 5. Heatmaps of predicted and actual labels on final submission results.

DistilBERT appears to have overfitted the train data on which it achieved very high F1-scores, but failed to perform well on the test data in the final submission. Our stacking method also failed to achieve high results in the final submission, being prone to predict "fake" news as can be seen in Fig. 5. On the other hand, the taxonomy based tax2vec data enrichment method as well as the LSA model have both shown good results in the final submission, while our best performing model used stacking, where we merged different neural and non-neural feature sets into a novel representation. With this merged model, we achieved 0.972 F1-score and ranked 50th out of 168 submissions.

In Fig. 5 we present the confusion matrices of the models evaluated in the final submissions.

7 Conclusion and Further Work

In our take to tackle the detection of fake-news problems we have have exploited different approaches and techniques. We constructed hand crafted features that captured the statistical distribution of words and characters across the tweets. From the collection of n-grams of both character and word-based features to be found in the tweets we learned a latent space representation, potentially capturing relevant patterns. With the employment of multiple BERT-based representations we captured the contextual information and the differences between fake and real COVID-19 news. However such learning showed that even though it can have excellent results for other tasks, for tasks such as classification of short texts it proved to fall behind some more sophisticated methods. To overcome such pitfalls we constructed two different meta models, learned from the decisions of simpler models. The second model learned a new space from the document space representations of the simpler models by embedding it via a 5 layer neural network. This new space resulted in a very convincing representation of this problem space achieving F1-score of **0.9720** on the final (hidden) test set.

For the further work we suggest improvements of our methods by the inclusion of background knowledge to the representations in order to gain more instance separable representations. We propose exploring the possibility of adding model interpretability with some attention based mechanism. Finally, as another add-on we would like to explore how the interactions in the networks of fake-news affect our proposed model representation.

Acknowledgements. The work of the last author was funded by the Slovenian Research Agency (ARRS) through a young researcher grant. The work of other authors was supported by the Slovenian Research Agency core research programme *Knowledge Technologies* (P2-0103) and the ARRS funded research projects *Semantic Data Mining for Linked Open Data (ERC Complementary Scheme, N2-0078)* and *Computer-assisted multilingual news discourse analysis with contextual embeddings - J6-2581).* The work was also supported by European Union's Horizon 2020 research and innovation programme under grant agreement No 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media).

References

1. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
2. Dumais, S.T.: Latent semantic analysis. Ann. Rev. Inf. Sci. Technol. **38**(1), 188–230 (2004). https://doi.org/10.1002/aris.1440380105, https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/aris.1440380105
3. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions (2009)
4. Jwa, H., Oh, D., Park, K., Kang, J.M., Lim, H.: exBAKE: automatic fake news detection model based on bidirectional encoder representations from transformers (BERT). Appl. Sci. **9**(19), 4062 (2019)
5. Koloski, B., Pollak, S., Škrlj, B.: Multilingual detection of fake news spreaders via sparse matrix factorization. In: CLEF (2020)
6. Loper, E., Bird, S.: NLTK: the natural language toolkit. In: Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1, pp. 63–70. ETMTNLP 2002, Association for Computational Linguistics, USA (2002). https://doi.org/10.3115/1118108.1118117
7. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Guyon, I., et al., (eds.) Advances in Neural Information Processing Systems 30, pp. 4765–4774. Curran Associates, Inc. (2017). http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

8. Martinc, M., Skrlj, B., Pollak, S.: Multilingual gender classification with multi-view deep learning: notebook for PAN at CLEF 2018. In: Cappellato, L., Ferro, N., Nie, J., Soulier, L. (eds.) Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum, Avignon, France, 10–14 September 2018. CEUR Workshop Proceedings, vol. 2125. CEUR-WS.org (2018). http://ceur-ws.org/Vol-2125/paper_156.pdf
9. Ji, L., Wang, Y., Shi, B., Zhang, D., Wang, Z., Yan, J.: Microsoft concept graph: mining semantic concepts for short text understanding. Data Intell. **1**, 262–294 (2019)
10. Patwa, P., et al.: Overview of constraint 2021 shared tasks: detecting English covid-19 fake news and Hindi hostile posts. In: Chakraborty, T., Shu, K., Bernard, R., Liu, H., Akhtar, M.S. (eds.) CONSTRAINT 2021, CCIS 1402, pp. 42–53. Springer, Cham (2021)
11. Patwa, P., et al.: Fighting an infodemic: Covid-19 fake news dataset. arXiv preprint arXiv:2011.03327 (2020)
12. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
13. Princeton University: About wordnet (2010)
14. Reimers, N., Gurevych, I.: 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, November 2019. https://arxiv.org/abs/1908.10084
15. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. CoRR 1910.01108 (2019)
16. Shannon, P., et al.: Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Res. **13**(11), 2498–2504 (2003)
17. Shao, C., Ciampaglia, G.L., Varol, O., Yang, K.C., Flammini, A., Menczer, F.: The spread of low-credibility content by social bots. Nature Commun. **9**(1), 1–9 (2018)
18. Shu, K., Bernard, H.R., Liu, H.: Studying fake news via network analysis: detection and mitigation. In: Agarwal, N., Dokoohaki, N., Tokdemir, S. (eds.) Emerging Research Challenges and Opportunities in Computational Social Network Analysis and Mining. LNSN, pp. 43–65. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-94105-9_3
19. Škrlj, B., Martinc, M., Kralj, J., Lavrač, N., Pollak, S.: tax2vec: constructing interpretable features from taxonomies for short text classification. Comput. Speech Lang. **65**, 101104 (2020). https://doi.org/10.1016/j.csl.2020.101104, http://www.sciencedirect.com/science/article/pii/S0885230820300371