Using Artificial Neural Networks to Determine Ontologies Most Relevant to Scientific Texts

Lukáš Korel[1], Alexander S. Behr[2], Norbert Kockmann[2] and Martin Holeňa[1,3,4]

[1]Faculty of Information Technology, CTU, Prague, Czech Republic
[2]Faculty of Biochemical and Chemical Engineering, TU Dortmund University, Germany
[3]Institute of Computer Science, Czech Academy of Sciences, Prague, Czech Republic
[4]Leibniz Institute for Catalysis, Rostock, Germany

Abstract
This paper provides an insight into the possibility of how to find ontologies most relevant to scientific texts using artificial neural networks. The basic idea of the presented approach is to select a representative paragraph from a source text file, embed it to a vector space by a pre-trained fine-tuned transformer, and classify the embedded vector according to its relevance to a target ontology. We have considered different classifiers to categorize the output from the transformer, in particular random forest, support vector machine, multilayer perceptron, k-nearest neighbors, and Gaussian process classifiers. Their suitability has been evaluated in a use case with ontologies and scientific texts concerning catalysis research. From results we can say the worst results have random forest. The best results in this task brought support vector machine classifier.

Keywords
ontology, text data, text preprocessing, text representation learning, text classification

1. Introduction

A domain ontology defines a set of representational primitives with which to model a domain of knowledge or discourse. The representational primitives are typically classes, attributes, and relationships. The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application. Classes can be defined in two ways: by annotating their definitions, or by connecting classes with each other and with properties. Each domain ontology typically uses domain-specific definitions of terms denoting its primitives.

The FAIR research data management (Findable, Accessable, Interoperable, and Reuseable) needs a consistent data representation in ontologies, particularly for representing the data structure in the specific domain [34]. Since different ontologies are written by different people, they are often incompatible, even within the same domain. As systems that rely on domain ontologies expand, it is often needed to merge domain ontologies by manual tuning. The same is true for enhancing an ontology with information available in domain-related texts. Merging and enhancing ontologies is thus a largely manual process and therefore time-consuming and expensive.

ITAT'22: Information technologies – Applications and Theory, September 23–27, 2022, Zuberec, Slovakia
lukas.korel@fit.cvut.cz (L. Korel);
alexander.behr@tu-dortmund.de (A. S. Behr);
norbert.kockmann@tu-dortmund.de (N. Kockmann);
martin@cs.cas.cz,martin.holena@catalysis.de (M. Holeňa)

The need to find a suitable ontology for an input text can help in classifying the information presented within the text as well as to connect the input text with data. This would allow for automated selection of ontologies and respective classification of the text. Different text data could thus be compared automatically in an understandable way and connected with corresponding research data. Ontologies represent "a formal specification of a shared conceptualization" [7] and can thus be used to express knowledge and data in a formalized, standardized description language to specify terms and relations between those terms.

Current ontology recommenders, such as the NCBO ontology recommender [8], score annotations based on words similar to preferred and alternate labels of ontology classes and term frequency. In contrast to this, this work aims to use text representation learning in order to not only search for words also contained in ontologies but also to find concepts with similar semantic meaning between text and ontology.

This paper is devoted to a specific problem encountered during enhancing ontologies and sometimes during their merging: to decide which of several available ontologies is most relevant to given domain-related piece of text. Our solution to the problem relies primarily on artificial neural networks (ANNs), in particular on natural language processing (NLP).

The next section surveys the applicability of artificial neural networks to ontologies. Section 3 recalls the employed methods of text preprocessing. There have been used modules for text extractions from PDF files, for transforming extracted files to pure text and for eliminating irrelevant paragraphs. In the section is described text representation learning, as well as the principles of the employed classifiers. In section 4, an application of the proposed methodology to catalysis is described and evaluated.

With regard to sources we have studied described in part 2 of this article, we are not aware that classifiers learned from the results of representational learning have ever been used to determine the most relevant of a given set of ontologies.

2. Applicability of Artificial Neural Networks to Ontologies

In connection with learning and extending ontologies, artificial neural networks (ANNs) have been primarily used for identification of concepts, relations and attributes [10, 15, 18]. With respect to relations, some ANN-based methods have been developed specifically for subsumption relations needed for the construction of taxonomies [11, 14, 21, 30]. In connection with integration of ontologies, they have been primarily used for ontologies matching aka ontologies alignment [12, 13, 16, 33]. The variety of employed kinds of ANNs is rather large. It includes traditional multilayer perceptrons (MLPs) [19], adaptive resonance theory (ART) networks [17] and associative memories [23], as well as the modern deep convolutional networks (CNNs) [12, 20], deep belief networks [10], long short term memory (LSTM) networks together with their bidirectional variant (BiLSTM) [24] and gated recurrent units (GRU) networks [28, 29]. The dependence of ontologies on texts led to using networks developed for text and natural language representation learning, most importantly BERT [22, 26], the bidirectional encoder representations from transformers, and word2vec [25], the most traditional network for embedding text into an Euclidean space. The close relationship of ontologies to knowledge graphs led to using also RDF2Vec [21, 29], which was originally proposed for knowledge graphs [31]. In connection with word2vec and RDF2Vec, it is on similar principles, the network OWL2Vec was proposed for embedding of ontologies [32]. Finally, the graph-like structure of ontologies brought usage graph neural networks (GNNs) [16, 33].

Closest to the proposed project is the way ANNs have been recently used in connection with translating into OWL [27, 28], with predicate chaining and restriction [23], and with taxonomy extraction from knowledge graphs [21]. In [27], ontology learning is tailored as a transductive reasoning task that uses two recurrent neural networks to translate text in natural language into OWL specifications in description logic. That approach was further developed in [28], resulting in a system based on a single recurrent network of GRU type. It uses an encoder-decoder configuration and translates through syntactic transformation a subset of natural language into the description logic language ALLQ. Moreover, the system generalizes over different syntactic structures, and has the ability to tolerate unknown words through copying input words as extralogical symbols to the output, as well as the ability to enrich the training set with new annotated examples. In [23], a mapping is established between ontologies and a pair of interacting associative memories. One of them stores assertions, and the other stores entailment rules. The most recent work [21] describes a method for the specific task of extracting a taxonomy from an embedding of a knowledge graph. Over that embedding, which can be obtained for example with RDF2Vec, hierarchical agglomerative clustering is performed, first without using type information, and then injecting types into the hierarchical clustering tree. In addition, an axiom induction algorithm is applied to each cluster in the resulting tree, which allows to identify new classes corresponding to those axioms that describe their respective clusters accurately enough.

Neural networks are often used due to their strengths in natural language processing task. Ontology construction rely very much on texts, which suggest the applicability of artificial neural networks (ANNs) in this context.

3. Methodological Background

This section describes details of used methods to reach requested target. In the first part we need receive content from textural files, parse it into paragraphs and keep only paragraphs fulfilling minimal length and relevant content to the topic of the document. The second part describes usage selected transformer and embedding input paragraphs to classification numeric vectors. The final part describes used classifiers, which use outputs from the transformer for final classifications to target ontology.

3.1. Text Preprocessing

For the problem scientific texts classification to the most relevant existing ontology, we have been using documents in portable document files (PDFs). An issue with PDFs is that they are optimized to print on physical printer, thus they contain meta-information about the contained text related to the position on the page. Therefore, it is not easy to address a single paragraph. If the file is read using the basic library for PDF files and the newline mark is used as the splitter, it returns only a single row, not the whole paragraph. Another issue is connected with multi-column documents. If the document does not include information about where the text continues, software libraries for text extraction from PDF usually continue with the next letter on the same row.

One solution to get text data from multi-column PDF is to use Microsoft Word engine. Its engine is able to solve both problems and parse text properly. It identifies structural information in text such as headings, paragraphs and sentences. Each document may contain texts irrelevant to the topic of interest, for example references, acknowledgement etc.

Specifications of the ontologies are most often stored in OWL files. OWL [2] is a specific kind of XML for ontologies. Text that describes classes and relations may be stored in different tags, depending on the decision of the ontology designer.

3.2. Text Representation Learning

For typical data analysis tasks like classification of clustering, it is suitable to represent words or other parts of text by vectors in an Euclidean space. Such representation is mostly the result of representation learning by ANNs. In the area of text analysis and processing, the probably most successful representation learning algorithm is BERT (Bidirectional Encoder Representations from Transformers)[3].

BERT needs to be trained using large amount of texts. That is why some pretrained version is typically used, and often subsequently fine-tuned using texts concerning the considered topic. Such fine-tuning is often performed even if the pretrained network was trained, apart from general texts, also with texts from some broader relevant domain (biology, medicine, chemistry, etc.).

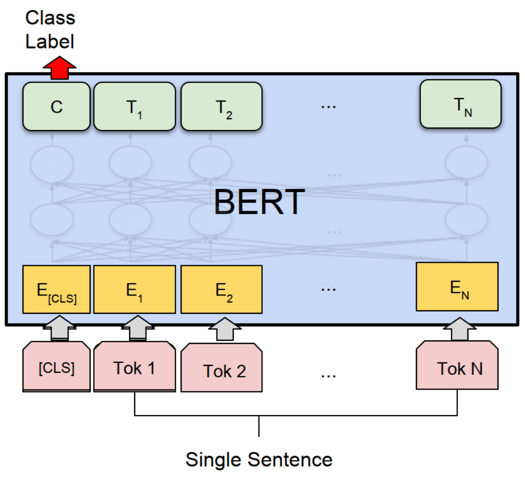

Figure 1: BERT (Bidirectional Encoder Representations from Transformers) architecture [3]. An input sentence is divided into tokens and each token is encoded to number. The BERT's output contains one numeric vector per one token. The output marked as C is used for final classification.

The basic schema of BERT is given in Figure 1. The tokenized input at first passes through the encoder, which embeds sentences to elements of an Euclidean space. These vectors are used as input to the BERT decoder. BERT returns one vector for each input. Each input sequence contains a special token at the beginning marked as CLS. Vectors embedding the tokens of an input sequence can be arranged into a matrix. The first row of the matrix is the embedding of the whole input. Details of BERT are described in [3] and on the https://huggingface.co/docs/transformers/model_doc/bert. These embeddings of every input paragraph are taken into account for the final assignment of the most relevant ontology to the paragraph.

3.3. Classification

The embeddings obtained in BERT are used as inputs for classifiers classifying a given input part of text (e.g., a paragraph) with respect to its relevance to the considered ontologies. Those classifiers have been trained on the embeddings of the annotations from the considered ontologies because for them, the ground truth (i.e., the ontology to which the annotation belongs) is known.

We have decided to select five classifiers implemented in scikit-learn [4]. They are the following:

1. Random forest (RF): An ensemble classifier that fits a number of classification trees on various sub-samples of the training data and uses some aggregation function to improve the predictive accuracy and control over-fitting. Usually, each tree in the ensemble is built using a sample drawn with replacement (i.e., a bootstrap sample) from the training set. Furthermore, when splitting each node during the construction of a tree, the best split is found using either all input features or a random subset of a given size. The purpose is to decrease the variance of the forest estimator. Indeed, individual decision trees typically exhibit high variance and tend to overfit. The injected randomness in forests yield decision trees with somewhat decoupled prediction errors. By taking an average of those predictions, some errors can cancel out. RFs achieve a reduced variance by combining diverse trees, sometimes at the cost of a slight increase in bias. Typically, the variance reduction yields an overall better model [35].
2. Support vector machine (SVM): It is a classifier designed specifically to achieve the lowest possible predictive error, using a known relationship between generalization error and margin of the separating hyperplane. It uses only training points on both support hyperplanes of the margin (support vectors), so it is also memory efficient. A

simple SVM can be used only for linearly separable classes. For linearly nonseparable classes, the data must be first transformed to linearly separable sets of functions in a high-dimensional vector space of functions using a suitable kernel. The SVM classification has multiclass support handled according to a one-vs-one or one-vs-rest scheme [36].

3. Gaussian Process (GP): It has been designed primarily for regression problems. A Gaussian Process Classifier (GPC) implements a collection of random variables indexed by an Eucliedan space for classification purposes through placing a GP prior on latent functions. Its purpose is to allow a convenient formulation of the classification through a logistic link function. GPCs support multi-class classification by performing either one-versus-rest or one-versus-one training and prediction. A crucial ingredient of each GPC is the covariance functions of the underlying GP. It encodes the assumptions on the similarity of Gaussian distributions corresponding to different points [37].
4. K nearest neighbors: Neighbors-based classification simply stores instances of the training data. A query point is assigned the data class which has the most representatives within the nearest neighbors of the point. The nearest neighbors classification can use uniform weights, that means, the value assigned to a query point is computed from a simple majority vote of the nearest neighbors. In some cases, it is better to weight the neighbors in such a way that nearer neighbors contribute more to the fit. For example, when an unknown point's class is computed from two nearest points and one of this two is nearer than second, in weighted case is result class same as the nearer point. The distance d between two points can be computed as: $d(x,y) = (\sum_{i=1}^{n} |x_i - y_i|^c)^{\frac{1}{c}}$, where n is the dimension of each point and $c \geq 1$, if $c = 1$, this is the Manhattan distance and in case $c = 2$, this is the Euclidean distance [38].
5. Multi-layer Perceptron (MLP): Given a set of features and a target, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, because between the input and the output layer, there can be one or more non-linear hidden layers. The input layer consists of a set of neurons representing the input features. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation, followed by a non-linear activation function. The output layer receives the values from the last hidden layer and transforms them into output values. The advantages of MLP are capability to learn non-linear models and capability to learn models in real-time (on-line learning). But the MLP with hidden layers have a non-convex loss function where there exists more than one local minimum. Therefore, different random weight initializations can lead to different validation accuracy. A MLP requires tuning a number of hyperparameters such as the number of hidden neurons, layers, and iterations. Moreover, it is sensitive to feature scaling [39].

4. Case Study in Catalysis

A catalyst is some chemical that is not consumed in the process of a chemical reaction. Using a catalyst in a chemical reaction usually allows said reaction to take place faster and allows for more moderate reaction conditions. Catalysis-based chemical synthesis is applied at roughly 90% of chemical processes in chemical industry. The scientific domain of catalysis is highly interconnected to other sciences and thus spans over many topics from material sciences to process design [5, 6].

4.1. Used Data

The texts that have been used for fine-tuning BERT, have been taken from scientific papers in catalysis. These articles have been by PowerShell script extracted to Microsoft Word documents. Thanks to its engine, paragraphs and titles are marked properly, so paragraphs with relevant texts have been extracted and with BERT embedding prepared for classification.

We conduct our experiments on a set of five ontologies from the chemical domain (Table 1) gathered within the NFDI4Cat project [34]. The ontologies NCIT, CHMO and Allotrope have a close connection to the chemical domain. However, according to their names, the chemical entities of biological interest (CHEBI) and the system biology ontology (SBO) are expected to be further away from the chemical domain. This does not hold necessarily true for the CHEBI as it describes a plethora of chemical entities, also relevant in the chemical and not only biological domain. The SBO was selected as it contains some general laboratory and computational contexts. It also can be seen as some kind of a test, whether the tools used can also identify ontologies not fitting to the text content.

Hence, these ontologies are classes to which classifiers assign new parts of text. The data have been divided into training and testing datasets in stratified proportion 1:1. The testing dataset has been divided into 20 disjoint subsets, assuming that disjointness is a sufficient condition for their independence. The training dataset have been under-sampled in order to mitigate overfitting during

Table 1
Types and counts of labels in the used OWL files

Ontology name	XML classes	Number of classes
Allotrope	Literal rdfs:comment rdfs:label	2773
NCIT	rdfs:comment rdfs:label	1169
SBO	Literal rdfs:comment rdfs:label	534
CHEBI	obo:IAO_0000115 rdfs:label	35067
CHMO	obo:IAO_0000115 rdfs:comment rdfs:label	2521

training part.

4.2. Experimental Setting

At first, the PDFs were transformed into Microsoft Word using PowerShell scripts. The output files have been processed by a python library for parsing docx files. As a result relevant paragraphs have been extracted for classification according to the most relevant ontology. The irrelevant paragraphs contained acknowledgement, references, titles and too short paragraphs (shorter than 100 letters) have been skipped.

The annotations in the specifications of given ontologies have been extracted using XML parser for python named BeautifulSoup. Extracted paragraphs have also been used for BERT fine-tuning. The chosen version of the BERT was recobo/chemical-bert-uncased from the Huggingface portal [9]. Using the fine-tuned BERT, every paragraph has been transformed into a 768-dimensional numeric vector.

The extraction of annotations from OWL files has been performed using a python XML parser. Individual annotations have been again embedded into the 768-dimensional vector space using the fine-tuned BERT.

For the employed classifiers, their implementations in ScikitLearn [4] has been used. The optimal values of hyperparameters of each classifier were determined using a 5-fold cross-validation applied to a grid-search with the grid values listed in Table 2. In order to mitigate overfitting, training data have been undersampled. Statistic computations have used the scipy, statsmodels and pingouin python libraries.

4.3. Comparison of Important Classifiers on Considered Ontologies

Summary statistics of the predictive accuracy of classifying all 20 testing datasets are in Table 3. The table is complemented with boxplots (Figure 2), where the following quality measures are presented for each classifier: accuracy, F1 score, precision and recall. The random forest classifier had the worst results of all experiments. Other models had significantly better results. The best accuracy had the Gaussian process, its mean accuaracy was 97.5 % with very low standard deviation.

The differences between the considered classifiers were tested for significance by the Friedman test. The basic null hypothesis that the mean accuracy for all 5 classifiers coincides was strongly rejected, with the achieved significance $p = 3.02 \times 10^{-12}$. For the post-hoc analysis, we employed the Wilcoxon signed rank test with two-sided alternative for all 10 pairs of the compared classifiers, because of the inconsistence of the more common mean ranks post-hoc test, as pointed out in [40]. For correction to multiple hypotheses testing, we used the Holm method. The results are given in Table 4, good results has Support vector machine and Gaussian process classifier.

4.4. Classification of Scientific Texts with respect to Relevant Ontologies

For this experiment, we had no ground truth as to which of the available ontologies is the most relevant for each considered paragraph of text. We employed two collections of scientific papers from the area of catalysis. The small one are papers dealing with the topic of methanation of CO2, it consists of 28 PDFs, from which we have extracted 1 485 relevant paragraphs. The large one is the digital archive of papers (co-)authored by scientists from the Leibniz Institute of Catalysis (with the exception of very few papers with read protection), it consists of 3 450 PDFs, from which we have extracted 144 490 relevant paragraphs. The BERT embeddings of those paragraphs were classified by the five trained classifiers. The confidence is probability over all classes, that source paragraphs fits into target class. Every paragraph can be classified to more than one target class with specific confidence. The sum of confidences of each paragraph is one. In this experiment were used models trained in previous experiment.

4.4.1. Results for the small dataset

Figure 3 shows how many paragraphs each classifier assigned to each ontology. The Gaussian process, k-nearest neighbor, MLP and SVM assigned almost all paragraphs to the NCIT ontology. The random forest is most uncer-

Table 2
Hyperparameters of individual classifiers that were determined through grid-search on combinations of considered values. In the column Selected are values, that have been selected using a random stratified 5-fold cross-validation applied to a grid-search with the Considered values

Classifier	Hyperparameter	Considered values	Selected
Random forest	maximal depth	{5, 7, 9, 11}	11
	criterion	{entropy, gini}	gini
	count of estimators	{5, 10, 15, 20, 25, 30}	20
	fraction of features used in each split	{0.5, 0.7}	0.5
	bootstrap samples	{false, true}	true
Support vector machine	slack trade-off constant (C)	{1, 10, 100, 1000}	100
	kernel type	{linear, radial basic}	radial basic
	kernel coefficient gamma	[0.001, 0.0001]	0.001
Gaussian process	kernel	{radial basic, dot product, mattern, rational quadratic, white kernel}	matern
	random state	{0, 50}	unapplicable
K nearest neighbors	number of considered neighbors	{1, 5, 9, 13, 17}	9
	weights	{uniform, distance}	distance
	algorithm	{auto, ball tree, kd tree, brute}	auto
	distance metric exponent	{1, 2, 3, 4, 5}	2
Multi-layer perceptron	random state	{0, 1}	0
	activation function	{identity, logistic, tanh, relu}	tanh
	optimizer	{lbfgs, sgd, adam}	adam
	hidden layer size	{1, 4, 16, 64}	4
	strength of L2 regularization term	{0.0001, 0.05}	0.05
	learning rate for weights update	{constant, adaptive}	constant

Table 3
Quality measures of the considered classifiers aggregated over all 20 testing datasets (mean [%] $\pm$ standard deviation [%]), where $Accuracy = \frac{TP+TN}{TP+FN+TN+FP}$, $Precision = \frac{TP}{FP+TP}$, $Recall = \frac{TP}{FN+TP}$ and $F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision+Recall}$

	Accuracy	F1	Precision	Recall
Gaussian process	97.46 ± 0.39	89.48 ± 1.38	85.70 ± 1.35	95.88 ± 1.21
K-nearest neighbor	96.66 ± 0.67	87.60 ± 2.41	84.36 ± 2.69	92.73 ± 2.04
Multi-layer perceptron	96.99 ± 0.67	87.84 ± 1.59	84.03 ± 1.54	94.97 ± 1.58
Random forest	94.63 ± 0.69	82.00 ± 2.29	76.30 ± 2.34	90.76 ± 2.58
Support vector machine	97.16 ± 0.53	88.72 ± 1.85	84.64 ± 1.89	95.85 ± 1.69

tain among all classifiers, assigning most paragraphs to the CHEBI ontology, but some pragraphs also to each of the remaining four.

Figure 4 uses instead of the count of class predictions their confidences. The confidence of the SVM and MLP is very high, whereas that of the Gaussian process and random forest is substantially lower. The k-nearest neighbors classifier has rather high confidence also.

In Figure 5, the margin between the confidence of the predicted ontology and the second highest class confidence is shown. Again, the highest values are achieved by the SVM and MLP, whereas the Gaussian process and random forest have only small margin between the predicted and second most confident ontology, and the k-nearest neighbor has quite a high margin, but not so high as SVM or MLP.

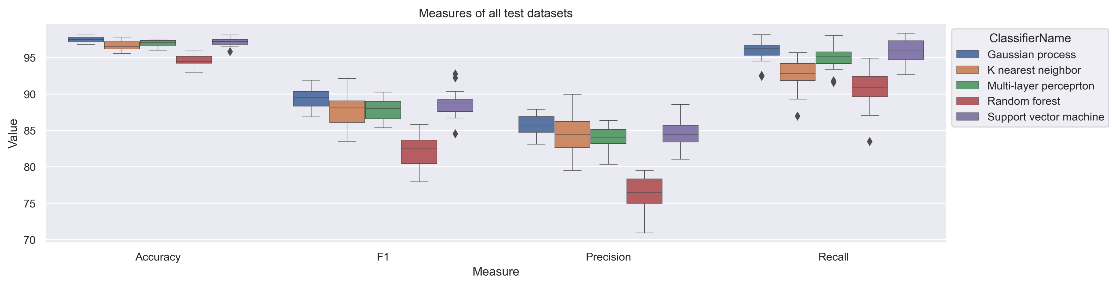

Figure 2: Box plots comparing the distribution of quality measures of the considered classifiers on testing datasets

Table 4
Comparison of accuracy results on all 20 testing sets with ontology annotations. The values in the table are counts of datasets, in which the model in the row has a higher accuracy compared to the model in the column. If the difference is not significant in the Wilcoxon test then the count is in italic. If the difference is significant, then the higher count is in bold.

	Random forest	Support vector machine	Gaussian process	K-nearest neighbors	Multi-layer perceptron	Summary score
Random forest	-	0	0	0	0	0
Support vector machine	**20**	-	2	**15**	*13*	50
Gaussian process	**20**	**16**	-	**17**	**19**	72
K-nearest neighbors	**20**	3	3	-	*5*	31
Multi-layer perceptron	**20**	*4*	1	*14*	-	39

4.4.2. Results for the large dataset

Figure 6 depicts the count of paragraphs from the large dataset that each classifier assigned to each ontology. The Gaussian process, k-nearest neighbor, MLP and SVM assigned almost all paragraphs to the NCIT ontology. The random forest is most uncertain among all classifiers, assigning most paragraphs to the CHMO ontology, but some pragraphs also to each of the remaining four.

Figure 7 using confidences of class predictions shows, that the confidence of the SVM is very high, whereas that of the Gaussian process and random forest is substantially lower. A rather high confidence have also the MLP and the k-nearest neighbors classifier.

In Figure 8, the margin between the confidence of the predicted ontology and the second highest class confidence is shown. Again, the highest values are achieved by the SVM, whereas the Gaussian process and random forest have only small margin between the predicted and second most confident ontology, and the MLP and k-nearest neighbor have quite a high margin, but not so high as SVM.

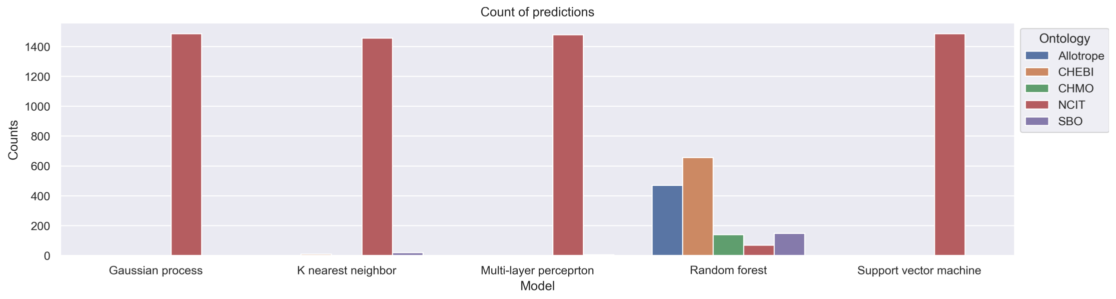

Figure 3: Counts of paragraphs of the small collection of scientific papers predicted by highest confidence to target class

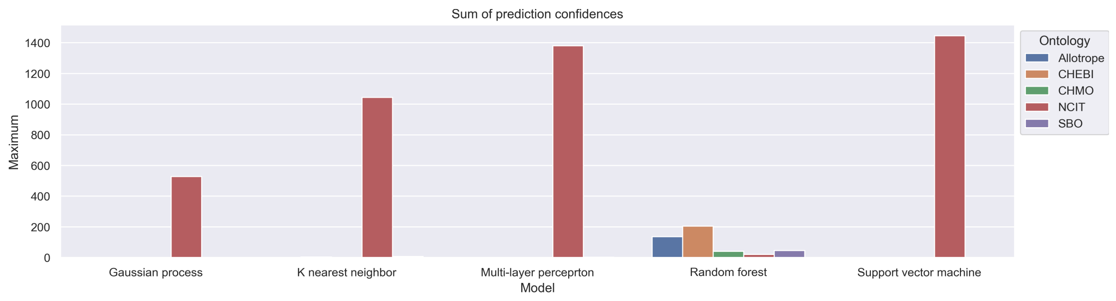

Figure 4: Sum of prediction confidences for the small collection of scientific papers

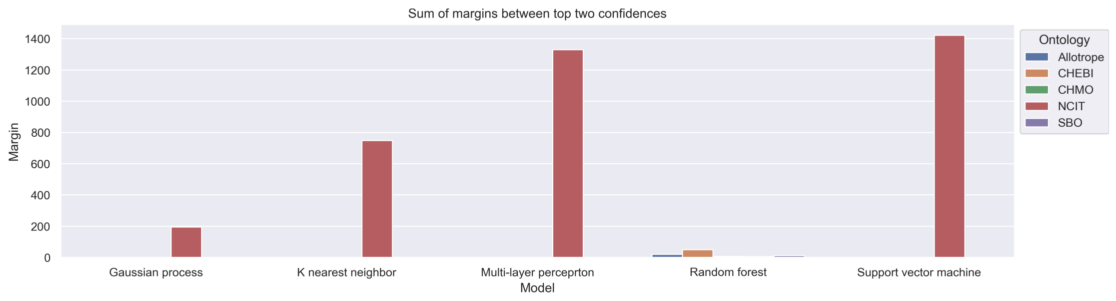

Figure 5: Sum of margins between top two confidences for the small collection of scientific papers

4.4.3. Summary results for both datasets

From results in the first experiment we can say SVM has good results on testing data in many metrics. The results for both datasets show that the SVM classifier has very high confidences and very high margins between top two confidences. Hence, the results indicate that for a large majority of the unknown scientific texts, the most relevant ontology is NCIT.

5. Conclusion

This paper provides an insight into the possibility to automatically determine ontologies most relevant to scientific texts. Successful processing input texts and ontologies often requires a quite hard and laborious job. Here have been used classifiers in combination with the representation learning by BERT, that may help make this process faster. Our idea was to use embedding of each paragraph from PDFs as input to classifiers. We used a pretrained BERT that have been fine-tuned using chemical articles. The output embeddings from fine-tuned BERT were used as an input to the classifiers. We have experimented with five different classifiers, in particular random forest, support vector machine, multilayer perceptron, k-nearest neighbors, and Gaussian process. The random forest was not successful, its accuracy was the worst of all models. The best results had Gaussian process and support vector

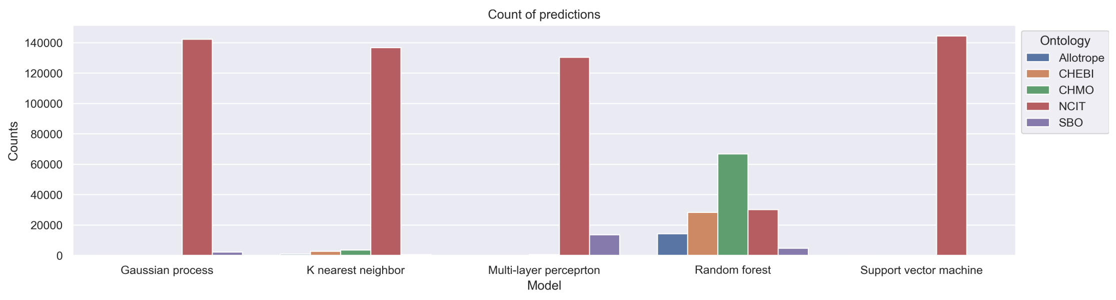

Figure 6: Counts of paragraphs of the large collection of scientific papers predicted by highest confidence to target class

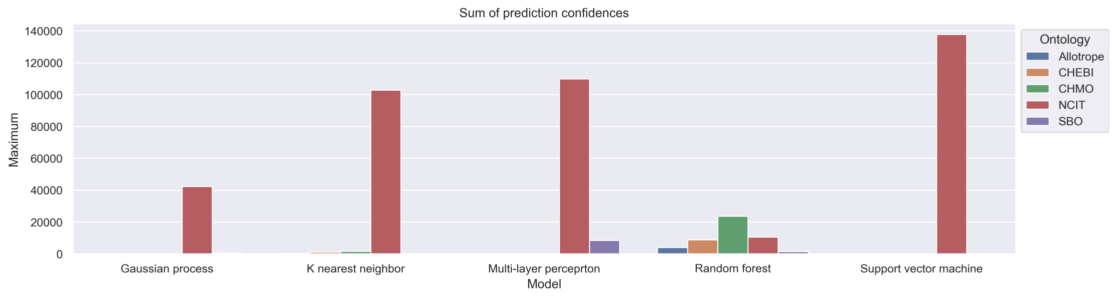

Figure 7: Sum of prediction confidences for the large collection of scientific papers

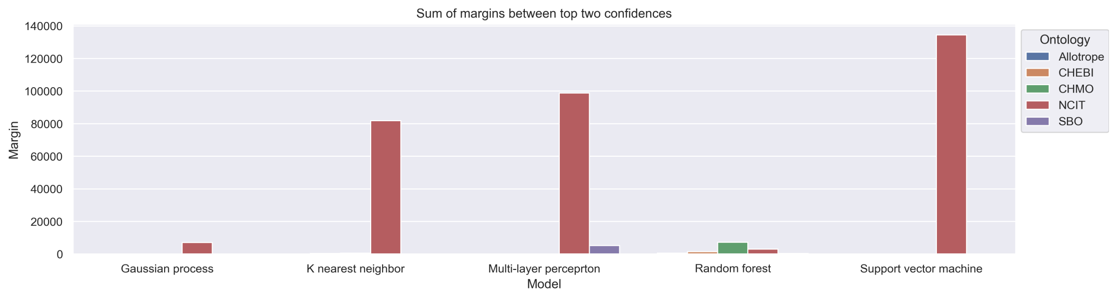

Figure 8: Sum of margins between top two confidences for the large collection of scientific papers

machine.

In second experiment the considered classifiers have been tested and compared on scientific papers from the domain of catalysis. The ground truth was not known there. The k-nearest neighbor and Gaussian process had very low margin between first and second highest confidence. The highest confidence among all classifiers had the support vector machine. It had also the highest margin among them.

The biggest weakness of this article is the lack of ground truth for the classification of scientific articles, which makes it impossible to evaluate this classification. Therefore, we plan to use methods for reducing the impact of unknown ground truth. Our idea is to use interpolation between annotations using GPT-2 and GPT-3 networks. GPT (Generative Pre-trained Transformer) [41] stands for a series of pre-trained language models, which have been developed by OpenAI. They have been trained with a large dataset of textual information and can be applied to deal with specific language-related tasks. BERT, which was trained with Wiki and books data that contains over 3.3 billion tokens, is popular in natural language understanding tasks, e.g., text classification. However, BERT as a masked language model can only learn contextual representation of words but not organize and generate language, which makes it unsuitable for design concept generation task. On the other hand, GPTs are autoregressive language models that are trained to predict the next token based on all tokens before it.

In future research, it is desirable to try different transformers. We would like to extract knowledge from ANNs in the context of learning. The main direction of our research is extending and integrating ontologies. We plan to use also graph neural networks to incorporate them into representation learning.

Acknowledgments

The research reported in this paper has been supported by the German Research Foundation (DFG) funded projects NFDI2/12020 and 467401796, and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS20/208/OHK3/3T/18.

References

[1] Gruber T., Liu L., Özsu M. T.: Ontology. `https://tomgruber.org/writing/definition-of-ontology.pdf` Encyclopedia of Database Systems, Springer-Verlag (2009)

[2] OWL Working Group: OWL. `https://www.w3.org/OWL/` W3C Semantic Web (2012-Dec-11)

[3] Devlin J., Chang M., Lee K., Toutanova K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding `https://aclanthology.`

org/N19-1423 Proceedings of the 2019 Conference of the North American Chapter: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics (June 2019), pp. 4171–4186

[4] Pedregosa et al.: Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research 12 (2011), pp. 2825–2830

[5] Benvenuto M. A., Plaumann H.: Industrial Catalysis (2021). Berlin, Boston: De Gruyter (De Gruyter STEM).

[6] American Chemical Society Report: Technology Vision 2020 The Chemical Industry, December 1996

[7] Borst W. N. Construction of Engineering Ontologies for Knowledge Sharing and Reuse Enschede Centre for Telematics and Information Technology (CTIT), 1997

[8] Martínez-Romero M., Jonquet C., O'Connor M. J., Graybeal J., Pazos A., Musen M. A.: NCBO Ontology Recommender 2.0: An enhanced approach for biomedical ontology recommendation Journal of biomedical semantics 8 (1), pp. 21 (2017)

[9] Recobo AI Company: BERT for Chemical Industry https://huggingface.co/recobo/chemical-bert-uncased HuggingFace - Transformers - BERT (2022)

[10] Al-Aswadi F. N., Chan H. Y., K.H. Gan: Extracting semantic concepts and relations from scientific publications by using deep learning. Proceedings of IRICT 2020, pp. 374—383, 2021.

[11] Althubaiti S., Kafkas S., Abdelhakim M., Hoehndorf R.: Combining lexical and context features for automatic ontology extension. Journal of Biomedical Semantics, 11:article no. 1, 2020.

[12] Bento A., Zouaq A., M. Gagnon: Ontology matching using convolutional neural networks. LREC, pp. 5648-–5653, 2020.

[13] Chakraborty J., Yaman B., Virgili L., Konar K., Bansal S. K.: OntoConnect: Results for OAEI 2020. OM ISWC, pp. 204-–210, 2020.

[14] Espinoza-Anke L., Ronzano F., Saggion H.: Hypernym extraction: Combining machine-learning and dependency grammar. CICLing, pp. 372-–383, 2015.

[15] Gupta N., Podder S., Annervaz K. M., Sengupta S.: Domain ontology induction using word embeddings. ICMLA, pp. 115—119, 2016.

[16] Hao L., Lei C., Efthymiou V., Quamar A., Özcan F., et al.: MEDTO: Medical data to ontology matching using hybrid graph neural networks. KDD'21, pp. 2946-–2954, 2021.

[17] Hourali M., Montazer G. A.: Using ART2 neural network and bayesian network for automating the ontology constructing process. Procedia Engineering, 29:3914—3923, 2012.

[18] Katyshev A., Anikin A., Denisov M., Petrova T.: Intelligent approaches for the automated domain ontology extraction. International Congress on Information and Communication Technology, pp. 410-–417, 2021.

[19] Kolozali S., Fazekas G., Barthet M., Sandler M. B.: A framework for automatic ontology generation based on semantic audio analysis. Audio Engineering Society International Conference, pp. 87-–96, 2014.

[20] Li G.: CNN based ontology learning algorithm and applied in PE data IAENG International Journal of Computer Science, 48:1-–8, 2021.

[21] Martel F., Zouaq A.: Taxonomy extraction using knowledge graph embeddings and hierarchical clustering SAC'21, pp. 836-–844, 2021.

[22] Memariani A., Glauer M., Neuhaus F., Mossakowski T., Hatings J.: Automated and explainable ontology extension based on deep learning: A case study in the chemical domain. 3rd International Workshop on Data Meets Applied Ontologies, pp. 1-–16, 2021.

[23] Mercier C., Chateau-Laurent H., Alexandre F., Viéville T.: Ontology as neuronal-space manifold: Towards symbolic and numerical artificial embedding Workshop on Knowledge Representation for Hybrid and Compositional AI, pp. 1—11, 2021.

[24] Mueller R. M., S. Abdullaev: Deep cause: Hypothesis extraction from information systems papers with deep learning for theory ontology learning Annual Hawaii International Conference on System Sciences, pp. 6250-–6259, 2019.

[25] Teslya N., Savosin S.: Matching ontologies with Word2Vec-based neural network ICCSA, pp. 745—756, 2019.

[26] Oba A., Paik I., Kuwana A.: Automatic classification for ontology generation by pretrained language model International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems, pp. 210—221, 2021.

[27] Petrucci G., Ghindini C., Rospocher M.: Ontology learning in the deep EKAW, pp. 480-–495, 2016.

[28] Petrucci G., Rospocher M., Ghindini C.: Expressive ontology learning as neural machine translation Journal of Web Semantics, pp. 52-–53:66-–82, 2018.

[29] Potoniec J.: Learning OWL 2 property characteristics as an explanation for an RNN Bulletin of the Polish Academy of Sciences, Technical Sciences, pp. 68:1481-–1490, 2020.

[30] Navarro-Almanza R., Juárez-Ramírez R., Castro J. R.: Automated ontology extraction from unstructured texts using deep learning Intuitionistic and Type-2 Fuzzy Logic Enhancements in Neural and Optimization Algorithms: Theory and Applications, pp. 727—755. Springer, 2020.

[31] Ristoski P., Paulheim H.: Rdf2vec: Rdf graph embeddings for data mining, International Semantic Web Conference, pp. 498—514, 2016.

[32] Ritchie A., Chen J., Castro L. J., Rebholz-Schuhmann D., Jiménez-Ruiz E.: Ontology clustering with OWL2Vec DeepOntoNLP, pp. 54-–61, 2021.
[33] Wu J., Lv J., Guo H., Daeom S. M.: A deep attentional embedding approach for biomedical ontology matching Applied Sciences, 10:article no. 7909, 2020.
[34] Wulf C., Beller M., Boenisch T., Hanf S., Deutschman O., and others: Research – Challenges and Concepts: A Unified Research Data Infrastructure for Catalysis ChemCatChem, pp. 3223–3236, vol 13, 2021.
[35] Ho T. K.: Random decision forests Proceedings of 3rd International Conference on Document Analysis and Recognition, 1995, pp. 278–282 vol.1, doi: 10.1109/ICDAR.1995.598994.
[36] Schölkopf B., Smola A. J.: Learning with Kernels, MIT Press 2002
[37] Rasmussen C. E., Williams C. K. I.: Classification in Gaussian Processes for Machine Learning MIT Press, 2005, pp.33–77.
[38] Kramer O.: K-Nearest Neighbors. In: Dimensionality Reduction with Unsupervised Nearest Neighbors (2013) Intelligent Systems Reference Library, vol 51. Springer, Berlin, Heidelberg, DOI: 10.1007/978-3-642-38652-7_2
[39] Vang-Mata R.: Multilayer Perceptrons: Theory and Applications Computer Science, Technology and Applications Series (2020) Nova Science Publishers, ISBN: 978-1-536-17365-9
[40] Benavoli A., Corani G., Mangili F.: Should We Really Use Post-Hoc Tests Based on Mean-Ranks? Journal of Machine Learning Research (2016), vol. 17, pp. 1–10
[41] Zhu Q., Luo J.: Generative pre-trained transformer for design concept generation: An exploration. Proceedings of the Design Society, 2: pp. 1825—1834, 2022.