Automatic Generation of Common Procurement Vocabulary Codes

Lucia Siciliani[1], Emanuele Tanzi[1], Pierpaolo Basile[1] and Pasquale Lops[1]

[1]University of Bari Aldo Moro, Department of Computer Science, via E. Orabona, 70125, Bari, Italy

Abstract

The role of tenders as means of investment of public funds and as vehicles of strategic development is nowadays crucial. For this reason, developing and enabling new solutions for e-procurement procedures can help to manage and invest funds. In e-procurement, the Common Procurement Vocabulary (CPV) allows assigning a code that classifies its subject to each tender. This study addresses the challenge of automatically assigning a CPV code to a tender. We tackle this problem in two different ways: as a classification problem and as a generative task. To develop and test our models, we build a dataset of 5M Italian tenders extracting them from the National Anti-Corruption Authority (Autorità nazionale anticorruzione - ANAC) website. Results show that text classifier approaches exhibit superior performance in this regard. However, they also reveal the potential of generative models in overcoming the limitations of existing classification methods for CPV code assignment in tender classification, providing valuable insights for improving procurement processes and enhancing efficiency in public sector operations.

Keywords

Natural Language Processing, e-procurement, e-tendering, Text Classification

1. Introduction

Knowledge organization systems (KOS), such as thesauri, gazetteers, lexical databases, ontologies, and classification systems, are used by institutions to organize large data collections, e.g. documents, web pages, and texts. Using a standard format guarantees semantic interoperability and allows for a faster exchange of information. Public procurement represents a field where adopting such systems can bring many advantages. On one hand, citizens can access data more easily, enabling more straightforward communication with institutions, which can help streamline many bureaucratic processes. Concurrently, adopting KOS systems in procurement has profound implications for professionals working within public administrations. In fact, these systems can serve as invaluable tools, offering support in the day-to-day activities of public sector employees. Integrating advanced technologies will facilitate a paradigm shift towards higher productivity and efficiency. Tasks that were once labor-intensive and time-consuming can now be executed with greater precision and speed, allowing public administrators to focus on more strategic and value-driven aspects of their roles. For this reason, in the field of public procurement, the European Union developed a Common Procurement Vocabulary (CPV)[1] that identifies the subject of a tender. The adoption of the CPV also allows companies to find new public contracts easily, thus fostering competitiveness.

The CPV is structured as a tree of codes comprising 9 digits, eight plus a check digit, and specifies whether the tender in question refers to supplies, works or services covered by the contract. Each digit indicates progressively finer-grained classifications. More specifically, each CPV is composed as follows:

- the first two digits identify the divisions (e.g. 71000000-8 Servizi architettonici, di costruzione, ingegneria e ispezione (Architectural, construction, engineering and inspection services));
- the first three digits identify the groups (e.g. 71300000-1 Servizi di ingegneria (Engineering services));
- the first four digits identify the classes (e.g. 71310000-4 Servizi di consulenza ingegneristica e di costruzione (Consultative engineering and construction services));
- the first five digits identify the categories (e.g. 71311000-1 Servizi di consulenza in ingegneria civile (Civil engineering consultancy services));
- each of the last three digits provides an additional degree of precision within each category (e.g. 71311210-6 Servizi di consulenza stradale (Highways consultancy services));
- a ninth digit serves to verify the previous digits.

Examples of CPV codes are: *30200000-1 (Computer equipment and supplies)*, *30230000-0 (Computer hardware)*, and

[1]https://simap.ted.europa.eu/it/web/simap/cpv

CLiC-it 2023: 9th Italian Conference on Computational Linguistics, Nov 30 — Dec 02, 2023, Venice, Italy

lucia.siciliani@uniba.it (L. Siciliani); e.tanzi2@studenti.uniba.it (E. Tanzi); pierpaolo.basile@uniba.it (P. Basile); pasquale.lops@uniba.it (P. Lops)

0000-0002-1438-280X (L. Siciliani); 0000-0002-0545-1105 (P. Basile); 0000-0002-6866-9451 (P. Lops)

© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3596/paper45.pdf

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

S*30231000-7 (Computers and printers)*.

The supplementary vocabulary can be used to complete the description of the subject of a contract. The items consist of an alphanumeric code corresponding to a denomination that allows you to provide further details on the specific nature or destination of the asset to be purchased. The alphanumeric code is structured as follows:

- a first level, consisting of a letter corresponding to a section (e.g. A Materiali (Materials));
- a second level, consisting of a letter corresponding to a group (e.g. AA Metalli e leghe (Metal and alloy));
- a third level, consisting of two digits corresponding to the attribute (AA02-4 Alluminio (Aluminium));
- the last digit is used to verify the previous ones.

Examples of supplementary codes are the following: *AA01-1 Metal*, or *UB05-6 Office items*.

The main vocabulary comprises 9,454 terms and, more specifically, 45 divisions, 272 groups, 1,002 classes, 2,379 categories, and 5,756 sub-categories. Assigning a CPV to a tender is a task which is accomplished by RUPs (Responsabile Unico del Procedimento, i.e. Tender's Managers), however, given the high number of terms, it is really difficult even for human experts to identify the right CPV to use. For this reason, despite assuring a fine-grained classification, the high number of labels frequently leads to errors in the CPV assignment like typos or wrong interpretation of the description of each code. Another phenomenon is represented by the skewed usage of the codes as there is a small number of CPVs which are more known and thus used more frequently while a large number of CPVs are underused. Given these premises, in this work, we propose a method for automatically classifying tenders to their CPV codes.

The paper is organized as follows: Section 2 provides an overview of the approaches available at the state of the art, Section 3 contains the details of the proposed solution, Section 4 reports the results obtained by the evaluation of our model, and finally Section 5 closes the paper.

2. Related Work

The assignment of a Common Procurement Vocabulary (CPV) code to a tender is crucial for the accurate identification and precise retrieval of analogous documents. This meticulous categorization process is indispensable for ensuring the streamlined organization and effective utilization of information. Given the nature of the CPV vocabulary, which encompasses a set of over nine thousand terms, this task assumes a challenging dimension, demanding a level of expertise that even seasoned professionals in the field find daunting. Already existing approaches for CPV classification have been condensed within the last few years. In [1], the author compared different deep-learning models for single-label and multi-label CPV classification. The best-performing model was represented by GRU [2] with attention mechanism [3]. The dataset used in this work comprises 30,000 Swedish tenders provided by e-Avrop[2], a Swedish company that manages an e-procurement platform. Each document in the dataset comprises titles and descriptions of each tender with their respective categories.

In [4], the authors used a dataset of 40,000 tenders extracted from TED. The authors have used an LSTM [5] architecture for sequence prediction and classification. They also used a Support Vector Machine (SVM) to classify the CPV main code category within the same framework. The PhD thesis by [6] addressed the CPV classification problem using a Linear SVM and a bag of words representation from a random sample of 200,000 documents extracted from the TED. An important aspect to notice is that this work focuses on both English and French. Kaan Görgün (Mkaan)[3] proposed a multilingual approach which is a fine-tuned version of mBERT [7] on tenders extracted from the TED. With their work, [8] are instead focused on the Spanish language. The proposed method uses RoBERTa-base-bne [9], a RoBERTa model pre-trained on Spanish documents. The authors fine-tune this model on Spanish Public Procurement documents, classifying the 45 CPV divisions. Next, they compare several models, ranging from more classical ones (e.g. Naive-Bayes, SVM, KNN, etc.) to the one proposed by MKaan.

Data from ANAC are a valuable resource for building data-driven systems in the public administration domain. In [10], authors propose an information extraction framework for Italian tenders [10] that leverage ANAC datasets and other information sources. Moreover, a decision support system that helps users during the entire course of investments and contracts in e-procurement is described in [11].

3. Methodology

The main idea is to support RUPs in assigning CPV to a new tender. Specifically, the aim is to establish a robust system capable of proficiently classifying a tender based on its specified object, thereby facilitating an accurate alignment with the comprehensive set of CPV codes available. The classification task is difficult since the number of codes (CPVs) is high. Therefore, we propose two

[2] https://info.e-avrop.com

[3] https://huggingface.co/MKaan/multilingual-cpv-sector-classifier

methodologies: 1) a text classification approach based on different classifiers; 2) a generative approach based on Transformers with an encoder-decoder architecture.

Both approaches work on the same data. In particular, given a list of tuples *(CPV, CPV description, tender object)*, we split it into three sets: training, validation and testing. More details on the dataset are reported in Section 4. Then each approach is implemented, trained and validated separately on the same data.

3.1. Text Classification

Regarding text classification, our approach aligns with a classical pipeline:

- **Preprocessing**: the initial step involves the preprocessing of the tender object, wherein transformations such as lowercase conversion and tokenization are applied. These essential preprocessing techniques lay the groundwork for subsequent stages by standardizing the textual data;
- **Feature Vector Generation**: following the preprocessing step, we construct feature vectors for each tender object. This involves the utilization of both Bag-of-Words (BoW) and TF-IDF (Term Frequency-Inverse Document Frequency) approaches. By encoding the textual information into numerical representations, we aim to capture the salient features that contribute to the classification task;
- **Classifier Training and Tuning**: subsequently, a classifier is trained using the feature vectors generated in the previous step. The training process is complemented by the optimization of hyperparameters. This optimization is achieved with the use of a validation set, ensuring that the classifier retains generalization capabilities;
- **Evaluation**: the final stage of our classification pipeline involves the evaluation of the trained classifier on an independent test set. This allows an assessment of the model's ability to generalize and classify unseen tender objects. It serves as a critical benchmark to validate the effectiveness and robustness of the entire text classification system.

For the implementation, we rely on spaCy for text processing and scikit-learn for classification. The hyperparameters are found through the grid search. After a first evaluation, we select the following classifiers: Linear SVC and Multinomial Naive Bayes. Moreover, we cast the problem only to classify the divisions that are composed of 45 classes since the model cannot provide reasonable results when the whole set of CPVs is involved.

Moreover, we choose to investigate a classifier based on BERT using its tokenizer. Again, in this setting, we use only the divisions as labels to obtain reasonable results. This strategic decision is driven by recognising that achieving reasonable results across the entire spectrum of CPV codes poses significant challenges. By concentrating on this subset, we optimize the model's capacity to provide meaningful and accurate predictions within a more manageable scope.

3.2. Generative Approach

Since our approach aims to suggest a CPV given a tender's description, we decided to investigate the ability of AI generative methods to automatically produce a text given a textual input. Moreover, we want to test if a generative approach can provide better results when a large number of classes is involved, as in our domain. We adopt a classical encoder-decoder architecture that has proven to provide promising results in several NLP tasks. Encoder-decoder architectures are well-suited for solving sequence-to-sequence problems like machine translation, as they can effectively process variable-length input and output sequences. In this architecture, the encoder takes in a sequence of any length and converts it into a fixed-shaped state. On the other hand, the decoder maps the encoded state, which has a fixed shape, back to a sequence of variable length.

In our case, the encoder's input is the tender's object description and the decoder output is the CPV code and its description. It is important to underline that this method can produce a CPV code or a description that is not present in the original list of CPVs, while a text classifier produces as output a CPV from the set of predefined CPVs. This poses problems in the evaluation phase as comparing the output produced with the gold standard present in the test set becomes more complex. More details about the evaluation are reported in Section 4.

4. Evaluation

For building and testing, we extract data from the ANAC anticorruzione (*anticorruption*) website[4]. ANAC -Autorità Nazionale AntiCorruzione *National Anti-Corruption Authority* is an Italian independent administrative authority with the aim of combating corruption in the country.

In particular, we retrieve for each tender the CIG (the tender identifier), the tender type, the description of the tender object, the CPV assigned by the RUP and the CPV description. The tender type identifies three kinds of tender: 1) supplying, 2) service and 3) work. From the original dataset, we remove CPV codes that occur less than 20 times and store data in a CSV file for a total of 5 million tenders. We split the dataset in training, test

[4] https://dati.anticorruzione.it/opendata/dataset/

and validation according to the percentages reported in Table 1.

	size	%
training	3,200,000	64%
test	1,000,000	20%
validation	800,000	16%

Table 1
Dataset statistics.

For training the encoder-decoder architectures, we build a different version of the dataset in the JSONL format. Each row in the dataset is a JSON object with two elements: *source* and *target*. The *source* is the input text of the encoder and the *target* is the output text of the decoder. In our case, the *source* is the concatenation of the type of the tender and the description of the tender's object, while the *target* is the concatenation of both the code and the description of the CPV. The tender's type defines the nature of the object from a list of predefined types: service, supply and work. Listings 1 shows an example of a JSON object related to a tender.

```
{"source":"lavori lavori di pavimentazione delle
     vie san martino e santa Maddalena",
"target":"45262321-7 - lavori di pavimentazione"
    }
```

Listing 1: An example of a JSON object for training the encoder-decoder architecture.

The dataset is stored on Zenodo[5], while the code is available on GitHub[6]. The code for fine-tuning the IT5 model is available here[7]. The IT5 models fine-tuned on the CPV generation task are on HuggingFace: the large model[8].

4.1. Text Classification

This sub-section reports information and results about text categorization approaches. Regarding the parameters' optimization, we adopt a grid search for Linear SVC and Multi-NB. For Linear SVC, we optimize the parameter C in the set {0.5, 1, 2, 4, 8}, while for Multi-NB we consider $alpha$ in {0.1, 0.3, 0.5, 0.7, 0.9} and fit_prior in {True, False}. The best values selected after the grid search are $C = 0.5$, $alpha = 0.3$ and $fit\_prior = False$.

For BERT, we did not perform parameters optimization since the required computational time is very high. We fine-tuned a specific language model for Italian called `dbmdz/bert-base-italian-uncased`[9] using the Adam optimizer with a learning rate of 5e-05 and a batch size of 16 trained for 5 epochs.

Results of text classification approaches are reported in Table 2. Generally, the results are very low due to the large number of classes and BERT reports the worst performance since, for some classes, there are very few examples in training data. We observe a large accuracy with respect to the F1 measure. This is due to the presence of few classes with many examples. For these classes, classifiers can achieve good performance. For example, BERT achieves the 90% of F1 for the most frequent class[10].

4.2. Generative Approach

We used the Java library Lucene for text searching and indexing, with the aim of solving the task as a retrieval task.

In detail, we indexed CPV code description pairs corresponding to *target* in the two fields *code* and *description*. Afterwards, we ran the search for each *source* element in JSON file, obtaining for each search the element belonging to target with the most similar description to the source string, and saved results in a file containing the triple *source*, *target* and *generated*. Where *target* is the expected description and *generated* is the retrieved one. We adopt the same output format for the generative approaches. The idea is to exploit the *source* as the query for the search engine and retrieve the most similar code descriptions using the search engine.

We executed this experiment four times, implementing variations in both the configuration of the text analyzer and the choice of two distinct similarity measures. The adopted configurations are the following:

- *StandardAnalyzer* + default similarity;
- *StandardAnalyzer* + *LMDirichletSimilarity*;
- *ItalianAnalyzer* + default similarity;
- *ItalianAnalyzer* + *LMDirichletSimilarity*.

The *ItalianAnalyzer* performs a specific stemming algorithm for Italian, while the *StandardAnalyzer* implements a grammar-based tokenizer for several languages. The default similarity provided by Lucene is the BM25 model [12], while the *LMDirichletSimilarity* uses a language model for information retrieval with the Bayesian smoothing based on Dirichlet priors [13]. The evaluation has been carried out on the test set. We decided to use BLEU metric to measure matching between generated and target text since this metric is based on the idea that the nearer the predicted text is to the target one, the more correct it is. Considering the small size of the compared strings, we decided only to use 1-gram and 2-gram of

[5] https://zenodo.org/records/10007545
[6] https://github.com/ematanzi/Valutazione-CPV/
[7] https://github.com/gsarti/it5
[8] https://huggingface.co/basilepp19/cpv-it5 and the base model https://huggingface.co/basilepp19/cpv-it5-base
[9] https://huggingface.co/dbmdz/bert-base-italian-uncased

[10] 33000000-0 Apparecchiature mediche, prodotti farmaceutici e per la cura personale (*Medical equipment, pharmaceuticals and personal care products*)

classifier	accuracy	P	R	macro-F1
Linear SVC	0.7768	0.5420	0.3758	**0.4041**
Multi-NB	0.7282	0.4334	0.3126	0.3292
BERT	0.7411	0.3136	0.2948	0.2990

Table 2
Results of text classification.

consecutive words, attributing them to the same weight (0.5). In calculating the metric, a smoothing function has been used, increasing the score when there are partial matches between the generated text and the target text.

In the evaluation, since the resolution of this classification task with utmost precision is arduous even for human experts, we decided to consider every possible correspondence between the code-description generated couple and the target one, which are the following:

- the full correspondence;
- the correspondence between codes, but not the description (the opposite case can never occur);
- the correspondence between categories;
- the correspondence between classes;
- the correspondence between groups;
- the correspondence between divisions;
- the case of no match.

In this way, we also evaluate the cases in which the solution has been approached. For all of these cases, we calculated the number of times they occurred along with the relative average of the BLEU score metric. Eventually, we also calculated the average BLEU score related to all the tests performed.

The best results obtained by the baseline are reported in Table 3. The results are obtained using the *ItalianAnalyzer* and the default similarity. The baseline based on the search engine is able to correctly retrieve the correct CPV with the correct description for only 11.34% of testing data. In the 61.77% of cases is not able to retrieve the correct CPV with very low BLEU (0.0515), this means that the description of the first retrieved CPV is very different from the correct one.

	# matchs	%	BLEU
Perfect match	113,440	11.34	1.0
Only CPV code	872	0.09	.5128
Only category	30,502	3.05	.2452
Only class	53,232	5.32	.1756
Only group	90,673	9.07	.1040
Only division	93,574	9.36	.1565
No match	617,707	61.77	.0515
All	-	-	.1751

Table 3
Best results obtained by Lucene with the *ItalianAnalyzer* and the default similarity.

We evaluate the generative approach based on the encoder-decoder transformer by fine-tuning it on training data. We start from a pre-trained Italian model called IT5 [14]. The IT5 model family is the initial endeavour to pre-train extensive sequence-to-sequence transformer models specifically designed for the Italian language, inspired by the methodology employed in the original T5 model [15]. We fine-tuned two different models with different sizes: IT5-lager and IT-base.

Results are reported in Table 4 for the large model and in Table 5 for the base one.

	# matchs	%	BLEU
Perfect match	372,188	37.22	1.0
Only CPV code	2,403	0.24	.4665
Only category	76,735	7.67	.2573
Only class	115,614	11.56	.2260
Only group	109,190	10.91	.1360
Only division	110,555	11.06	.1680
No match	213,315	21.33	.0861
All	-	-	.4546

Table 4
Results with the IT5-large model.

	# matchs	%	BLEU
Perfect match	359,408	35.94	1.0
Only CPV code	2,210	0.22	.4854
Only category	73,885	7.39	.2437
Only class	114,487	11.45	.2199
Only group	114,430	11.44	.1235
Only division	111,538	11.15	.1635
No match	224,042	22.40	.0853
All	-	-	.4385

Table 5
Results obtained with the IT5-base model.

To compare generative approaches with text categorization ones, we consider from the generated output only the first two digits of the CPV, i.e. the CPV divisions. This choice allows us to compare the generative approach with the ones based on text categorization since the latter are trained to predict only the division of each tender. Results of this analysis are reported in Table 6 and show that if we consider generative approaches as a classifier, they are below the simple Linear SVC. Performing a dual evaluation in which a classifier is evaluated as a generative approach is not possible since we train classifiers

only for predicting divisions. Considering only the code of the division is not possible to generate a description comparable to the text generated by an IT5 model.

model	acc.	P	R	macro-F1
Linear SVC	*0.7768*	*0.5420*	*0.3758*	*0.4041*
IT5-large	0.7867	0.4196	0.3575	0.3728
IT5-base	0.7760	0.4150	0.3525	0.3654

Table 6
Results of generative models evaluated as text classifier and compared with the best text categorization method.

Anyway, these outcomes are encouraging since the IT5 model is trained on all the possible descriptions of a CPV, while the text classifier approaches handle only the code of the division and cannot provide usable results when they are trained on the whole set of possible classes. Moreover, generative approaches provide a significant improvement with respect to the baselines obtained by a search engine.

5. Conclusions

In this paper, we tackle the challenge of categorizing a tender by aligning it with the comprehensive Common Procurement Vocabulary (CPV), i.e. a meticulously curated European lexicon of codes designed to precisely identify the subject matter of a tender. The complexity of this task lies in the diverse nature of procurement scenarios, where each tender has its own description and requirements. The CPV emerges as a fundamental tool in deciphering the procurement language, trying to define a European dictionary allowing interoperability among different countries. Our proposed methodologies encompass two distinctive approaches: the former relies on a conventional text classification paradigm, whereas the latter leverages a generative strategy hinging on the encoder-decoder architecture as conceptualized by the T5 model.

In our systematic exploration of the system's proficiency in discerning the accurate division of a tender, specifically on the initial two digits of the CPV, it becomes evident that text classifier approaches provide the best results. Nevertheless, a noteworthy result surfaces when we focus on the holistic identification of the entire CPV through a descriptive context. In this context, the generative approaches exhibit commendable efficacy, demonstrating promising outcomes. Notably, these generative techniques surpass established baselines constructed through conventional keyword-centric search engines, attesting to their heightened capabilities in nuanced comprehension and contextual inference.

Acknowledgments

We acknowledge the support of the PNRR project FAIR - Future AI Research (PE00000013), Spoke 6 - Symbiotic AI (CUP H97G22000210007) under the NRRP MUR program funded by the NextGenerationEU.

We thank Salvatore Tucci for his helpful support in evaluating text classification approaches.

References

[1] A. Suta, Multilabel text classification of public procurements using deep learning intent detection, Degree project in mathematics (second cycle), Kth Royal Institute of Technology, 2019.

[2] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078 (2014).

[3] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473 (2014).

[4] S. Kayte, P. Schneider-Kamp, A mixed neural network and support vector machine model for tender creation in the european union ted database., in: KMIS, 2019, pp. 139–145.

[5] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780.

[6] O. Ahmia, Assisted strategic monitoring on call for tender databases using natural language processing, text mining and deep learning, Ph.D. thesis, Université de Bretagne Sud, 2020.

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

[8] M. Navas-Loro, D. Garijo, O. Corcho, Multi-label text classification for public procurement in spanish, Procesamiento del Lenguaje Natural 69 (2022) 73–82.

[9] A. Gutiérrez-Fandiño, J. Armengol-Estapé, M. Pàmies, J. Llop-Palao, J. Silveira-Ocampo, C. P. Carrino, A. Gonzalez-Agirre, C. Armentano-Oller, C. Rodriguez-Penagos, M. Villegas, Spanish language models, arXiv preprint arXiv:2107.07253 (2021).

[10] L. Siciliani, E. Ghizzota, P. Basile, P. Lops, Oie4pa: open information extraction for the public administration, Journal of Intelligent Information Systems (2023). URL: https://link.springer.com/article/10.1007/s10844-023-00814-z. doi:https://doi.org/10.1007/s10844-023-00814-z.

[11] L. Siciliani, V. Taccardi, P. Basile, M. Di Ciano,

P. Lops, Ai-based decision support system for public procurement, Information Systems 119 (2023) 102284. URL: https://www.sciencedirect.com/science/article/pii/S0306437923001205. doi:https://doi.org/10.1016/j.is.2023.102284.

[12] S. Robertson, H. Zaragoza, The probabilistic relevance model: Bm25 and beyond, in: the 30th Annual International ACM SIGIR Conference, 2007, pp. 23–27.

[13] C. Zhai, J. Lafferty, A study of smoothing methods for language models applied to ad hoc information retrieval, in: ACM SIGIR Forum, volume 51, ACM New York, NY, USA, 2017, pp. 268–276.

[14] G. Sarti, M. Nissim, It5: Large-scale text-to-text pretraining for italian language understanding and generation, arXiv preprint arXiv:2203.03759 (2022).

[15] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, The Journal of Machine Learning Research 21 (2020) 5485–5551.