# Language Model CNN-driven similarity matching and classification for HTML-embedded Product Data*

Janos Borst[1][0000−0002−9166−4069], Erik Körner[1][0000−0002−5639−6177], Kobkaew Opasjumruskit[2][0000−0002−9206−6896], and Andreas Niekler[1][0000−0002−3036−3318]

[1] Leipzig University, Faculty of Mathematics and Computer Science, Institute of Computer Science, Augustusplatz 10, 04109 Leipzig, Germany
https://www.informatik.uni-leipzig.de/

[2] German Aerospace Center (DLR), Institute of Data Science, Mälzerstraße 3, 07745 Jena, Germany
https://www.dlr.de/content/en/institutes/institute-of-data-science.html

**Abstract.** The Semantic Web Challenge *Mining the Web of HTML-embedded Product Data* aims to benchmark current technologies on the data integration tasks (1) product matching and (2) product classification, as recent years have seen significant use of semantic annotations in the e-commerce domain, but often with inconsistencies, no complete coverage or conflicting information. We introduce a transformer-based approach for textual product matching and extend it with an CNN for product classification. We compare the influence of different input feature combinations against prediction performance and introduce a technique to augment the classification task with additional information. We are able to outperform baseline results using text-only approaches.

**Keywords:** product matching · product category classification · language models · natural language processing · text mining · deep learning

## 1 Introduction

The Semantic Web Challenge on Mining the Web of HTML-embedded Product Data declares two tasks, (1) product matching and (2) product classification, as main driver for product information integration services or research on product knowledge graph acquisition. The problem of data-driven automatic product data information emerged because semantic markup on the product information on the web is often sparse or inconsistent. Since there is no standard for product classification, and product vendors use their own category systems, third party product information integration services cannot rely on equal preconditions. As the main information page of the challenge [1] states correctly: "Addressing these challenges requires an orchestra of semantic technologies tailored to the product domain, such as product classification, product offer matching, and product

* Participating System in the Mining the Web of HTML-embedded Product Data

taxonomy matching. Such tasks are also crucial elements for the construction of product knowledge graphs, which are used by large, cross-sectoral e-commerce vendors." Because of this the challenge intends to assess the quality of systems addressing the two tasks. The challenge organizers developed data sets and resources, which realize the comparability of various approaches.

The definition of the shared task states product matching as a binary classification problem. Given two product descriptions, a system should decide whether they describe the same product or not. As mentioned before, product categorisations differ on different websites. The second task is therefore defined as classification of arbitrary product data sets into an unified single classification system. Our group addresses both tasks using language model driven neural classifiers.

In this paper we introduce a language model based approach for product similarity matching and an language model based multi output text classification network for product classification. The content of this work is structured as follows: In section 2 we position the task and methods we used to other related work. Section 3 will explain in detail the methods, architectures and data sets we used before presenting the results in section 4. We then conclude by discussing the results and pointing out possible improvements.

## 2 Related Work

The tasks we contribute to in this work are related to the fields of product classification, product matching and data linking. While the use of semantic annotations in the e-commerce domain has increased, it is still not sufficient in terms of consistency and completeness.

The similarity challenge of the product matching task is to predict, given a pair of structured product meta data, whether they describe the same product or not. Previous works on product classification, categorization and matching [19,12] perform well with text retrieval techniques and simple neural architectures and classification models like FastText [9] or Siamese Networks [23].

In the product classification domain, two similar data sets exist, i. e. Rakuten Data Challenge [3], which only deals with data gathered from a single source and the more closely related Web Data Commons [18] project, which is used as a basis for the data in this challenge.

The methods proposed in this paper are highly related to the field of natural language representation, text classification and text similarity. In recent years pre-training large language models have shown high impact on downstream tasks. Transformer models such as BERT [5] or RoBERTa [14] can be pre-trained on large amounts of data in an unsupervised fashion. The pre-trained models provide a numeric and context-sensitive representation of any text, which are then finetuned to a specific task using task-specific data. While earlier approaches based on word embeddings, like [17] or [15] often choose to keep text representations fixed during task-specific training, finetuning seems to be the core strength of the language model approach.

Text classification is a fundamental task in Natural Language Processing. Before language model finetuning became standard procedure, word embeddings combined with task-specific neural architectures provided state-of-the-art results in multi and single label classification [10,13,28]. In [10] a CNN-based architecture for text classification is presented, which exhibits robust results on a broad range of data sets. The CNN-layers extract features, which are then used to classify the text.

We hypothesize that textual similarity between product texts, like titles or descriptions, may be enough to decide for matching products. This bears structural similarity towards semantic textual similarity that was often topic of shared tasks [2,29,4] and has a variety of data sets [6,7]. It suggests that current transformer language models like BERT [5] that compete for SOTA scores in sentence pair classification are a good starting points for this task.

## 3 Classification Models and Data Flow

### 3.1 Task 1: Product Matching

**Sequence Pair Classification using Transformer Models** Our approach for the product matching task is based on the well known BERT [5] architecture as a good candidate to solve the product matching task using text features only. We use the Huggingface [26] implementations of the standard BERT model as well as RoBERTa [14] and their Distil* [20] variants with pre-trainend English language models, which we fine-tuned on the data sets. The model is structurally simple, it consists of a pre-trained transformer model which feeds its pooled output[3] into a dropout layer followed by a dense layer with either a single output for regression, or two outputs for classification ("same product" or not).

**Datasets and Features Usage** We chose to focus solely on the text features `title`, `description` and `specTable` of the product pair data[4] as they contained the most text content and were structurally more consistent compared to `keyValuePairs`, `brand` names or `prices`. We later show how those three features compare against each other and in combination. Depending on the choice of text features used, we simply concatenated them into a single sequence and annotated which sequence belonged to which product. No further text preprocessing steps were required as transformer models generally employ robust tokenizers, such as WordPiece [21,27] or Byte-Pair-Encoding [22], which can handle arbitrary text inputs. This resolves issues with unknown words.

In addition to the provided `computer` training and validation set, we also included the more exhaustive webdatacommons (WDC) product matching data

[3] The pooled output representation of BERT is based on the last hidden state of the [CLS] token, the first token in each sequence which is intended to learn information about the entire text sequence. For pooling, this output is fed through a dense layer with 768 units and tanh activation.

[4] An example of the data format can be found at: https://ir-ischool-uos.github.io/mwpd/index.html#task1

set [18] to have a wider variety of topics, more training and validation data (see Tab. 1) as well as a chance for better generalization.

| train set (attribute) | negative | positive |
|---|---|---|
| computer (title) | 58,771 | 9,690 |
| computer (desc) | 30,102 | 5,019 |
| computer (specTable) | 9,416 | 1,650 |
| WDC all | 184,462 | 30,198 |

Table 1: Training data set statistics, number of positive/negative product matching pairs per data set *computer* and *WDC all*, for *computer* also filtered by text feature occurrence.

### 3.2 Task 2: Product Classification

**Classification Model:** We employed a CNN architecture based on [10] for the product classification task. Since we understand the task as a single label multi output setting, we adjust the network to address this. As input to the network we use a transformer-based language model instead of static word vectors like GloVe [17] vectors. The core of the network is the CNN feature extraction layers, which we implement analogous to the original paper [10], but instead of one output layer, we use three, one for each hierarchy level of the data. A Dropout [24] layer is applied to the feature vector. For every output we calculate the loss using categorical crossentropy, which is then summed over all the outputs.

**External Data:** We support the training process by using the WDC [18] data set[5] and data extracted from Wikipedia. Since WDC uses the same category set as the task data, we can easily restrict it to the task's classes, which provides us with 8,004 additional examples.

Additionally, we also use generic descriptions derived from Wikidata [25] via its API. Names of classification examples from the training set are used to retrieve a set of candidate entities from Wikidata. We augment the descriptions from the training set with descriptions from Global Product Classification (GPC) standard [8] using the labels as references. These extended descriptions are used to disambiguate and filter relevant entities from the candidate sets using a tf-idf weight matrix. The entities are assigned to the label with the most similar context according to the training examples. From these entity sets, we construct training examples by joining the text content of the alternative labels, descriptions, common categories and summaries from the Wikipedia page provided by the Wikidata API. We use only retrieved entity descriptions for GPC level 3 and - since the GPC hierarchy is a tree - automatically assign the parent nodes. This process provides 1,394 additional training examples.

[5] English Goldstandard from http://webdatacommons.org/structureddata/2014-12/products/gs.html

## 4 Results

The organizers set baseline results for product matching with 90.8% F1 on the validation set using deepmatcher [16], and 85.734% Weighted Avg. F1 for the task product classification using FastText [9]. In what follows, we present our experiments on the validation sets and the final model configurations we used to submit to the official leaderboard. [6]

### 4.1 Task 1: Product Matching

**Training and Hyperparameters:** The *computer* training set shows a large bias towards the negative class (*"is not the same product"*, see Tab. 1), which we account for by using class weighted random sampling of the training data. We randomly discard about 80% of the negative product pairs in each epoch to match the number of positive samples and so avoid skewing the model towards negative predictions only.

We kept the default dropout of 0.1. The maximum sequence length of text input is a model dependent parameter, being either 128 or 512 tokens. Depending on the amount of text input, this leads to a truncation of the input. Correlating with the model dependent sequence length, we choose batch sizes of 8, 16 or 32, training for either 3 or 15 epochs. We also compare a two label ("matching" product or not) prediction setup using cross-entropy loss against a single output network using mean squared error loss (regression).

**Results:** We start with a simple BERT-base model approach and improve with more recent language models, combining various product text features, hyperparameter settings, and additional data. As shown in Tab. 2, starting from initially about 60%, we are able to increase the F1 by more than 30% on the computer validation set.

As shown in Tab. 2, the largest improvements stem from using the Distil* transformer model variants. Compared to BERT-base, they improve performance up to 25 percentage points, while consuming less memory. This makes either longer sequences or larger batch sizes possible. The distilled versions of RoBERTa further improve the F1 scores, although smaller in margin. Using the WDC product data corpus as additional training data only marginally improves results, indicating that the original data set is sufficient to finetune the computer topic and more generalization through other topics is not necessary.

In Tab. 3 we compare which text input feature combinations perform best while keeping other hyperparameters unchanged. As transformer models are not designed to artificially align input sequences consisting of differing features on some boundary and then pad them, we simply concatenate the text features into a single sequence for each product. The features `description` and `specTable` are sometimes empty, as shown in Tab. 1, and the various feature fields contain texts of varying lengths which results in differences of available contexts when

[6] https://ir-ischool-uos.github.io/mwpd/index.html#results

| model | epochs | train | eval | F1 |
|---|---|---|---|---|
| bert-base | 3 | comp | comp | 65.22 |
| bert-base | 3 | all | comp | 64.24 |
| distilroberta | 3 | all | comp | 91.73 |
| distilbert | 3 | all | comp | 87.62 |
| distilroberta | 3 | comp | comp | 91.41 |
| distilroberta | 15 | comp | comp | 95.05 |
| distilroberta (reg) | 15 | comp | comp | 95.57 |
| distilroberta | 15 | all | comp | **95.80** |
| distilroberta (reg) | 15 | all | comp | 95.00 |

Table 2: Overview of results for various hyperparameter configurations. All models used are uncased variants. Text input is a combination of `title+description+specTable`. *comp* being the computer only training set and *all* containing all categories. *reg* denotes regression instead of two class output.

| Features | P | F1 |
|---|---|---|
| title+description+specTable | **88.96** | 91.41 |
| title | 88.82 | **91.96** |
| description | 71.65 | 69.15 |
| specTable | 77.19 | 80.73 |
| description+title+specTable | 88.71 | 90.16 |

Table 3: Precision and F1-scores for label "matching product" on the computer train and validation sets using the DistilRoBERTa model, with 3 epochs of finetuning, and a sequence length of 512.

generating vector representations. However, the advantage of combining those text sequences is that more context is available for comparisons and that we can use alternative texts for possibly missing fields, e. g. descriptive `titles` and `description` texts.

We achieve the best results with 95.8% F1 on the computer validation set with the DistilRoBERTa-base model, using a sequence length of 512, a batch size of 16 and finetune for 15 epochs on the complete WDC categories training set (`all_gs.json`[7]). We combine the product text features `title + description + specTable` as a single input.

| Class | P | R | F1 |
|---|---|---|---|
| new products with high similarity with known products (25 pos / 75 neg) | 74.19 | 92.00 | 82.14 |
| new products with low similarity with known products (25 pos / 75 neg) | 63.16 | 96.00 | 76.19 |
| known products with introduced typos (100 pos) | 100.00 | 61.00 | 75.78 |
| known products with dropped tokens (100 pos) | 100.00 | 73.00 | 84.39 |
| very hard cases for known products (25 pos / 75 neg) | 91.67 | 88.00 | 89.80 |
| Overall result on hidden test set | 86.20 | 82.10 | 84.10 |

Table 4: Official analysis of submitted test predictions.

Manual inspection of false positives and false negatives in classified product pairs of the computer validation set show various edge cases like languages other than English, similar product attributes for different products etc. that are hard

[7] http://webdatacommons.org/largescaleproductcorpus/v2/index.html#toc6

to distinguish or match, even for humans. Tab. 4 is a detailed analysis on the "hidden" test set and proves that our model performs best on the set of edge cases ("very hard cases") in terms of F1 score, which are cases of highly similar negative pairs or highly dissimilar positive pairs. The sets of known products are both solved with a precision of 100 percent. This results in the highest precision of all systems in the competition.

### 4.2 Task 2: Product Classification

**Training and Hyperparameters:** As language model we employ DistilRoBERTa-base from the Huggingface library [26]. The model's weights can be fine-tuned during the supervised training. We use four CNN layers with kernel sizes of 3, 4, 5 and 6 with 100 filters each and a dropout rate of 0.5. The model is trained using the Adam [11] optimizer with a learning rate of 1 e -5 and a per label categorical cross-entropy. We pre-train our model on the WDC and/or Wikidata set for 20 epochs before switching to the task data. During training the model creates checkpoints each epoch and we report the results on the best epoch. From the task data we concatenated the text content of the following features: `name`, `description` and `url`.

**Results:** Since we chose a data driven approach we show the improvement each additional step brings to the base model:

- "Base": The BASE model denotes the proposed combination of DistilRoBERTa-base and multi output CNN architecture with a fixed language model.
- "FT": The language model weights are modified during training.
- "WDC": The model is pre-trained on the WDC data set.
- "Wiki": The model is pre-trained on the Wiki data set.

| | Average-P | Average-R | Average-F1 |
|---|---|---|---|
| Base | 73.02 | 76.13 | 72.76 |
| Base+FT | 88.91 | 88.51 | 88.36 |
| Base+FT+WDC | 93.64 | 92.79 | 92.93 |
| Base+FT+Wiki | 89.04 | 88.30 | 88.37 |
| Base+FT+WDC+Wiki | **93.83** | **93.48** | **93.39** |

Table 5: Results on the validation set of the product classification task averaged over per-level weighted average.

Tab. 6 shows the ablation study of every extension we added to the training. Unsurprisingly, the largest improvement stems from finetuning the model. When finetuning the model gains an order of magnitude in trainable parameters, going roughly from 1.5M to 83.6M parameters. The second big improvement stems from pre-training on the WDC data set. In a preliminary experiment we noticed that combining the task data and WDC resulted in worse results on the validation set. While pre-training on the Wiki data alone does not have a significant

impact on the final results, the combination of WDC and Wiki leads to the final model we use to predict on the test set. Tab. 6 breaks down the results per

| | Lvl1 | | | Lvl2 | | | Lvl3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 |
| Base | 80.17 | 81.10 | 79.21 | 76.70 | 78.97 | 76.17 | 62.19 | 68.33 | 62.89 |
| Base+FT | 91.49 | 91.30 | 91.24 | 90.76 | 90.50 | 90.43 | 84.48 | 83.73 | 83.40 |
| Base+FT+WDC | 95.53 | 95.00 | 95.13 | **95.03** | 94.33 | **94.48** | 90.36 | 89.03 | 89.17 |
| Base+FT+Wiki | 91.17 | 90.80 | 90.90 | 90.26 | 89.87 | 89.90 | 85.69 | 84.23 | 84.30 |
| Base+FT+WDC+Wiki | **95.56** | **95.37** | **95.33** | 94.63 | **94.53** | 94.40 | **91.31** | **90.53** | **90.43** |

Table 6: Weighted average of every classification level from the validation set of the product classification task.

level. Level 3 was the most difficult to predict, mainly stemming from the larger number of categories to classify. Here we see that the Wiki data seems to have a slight impact on the lvl3 categorisation, but worsens results in lvl1 and level 2, which may explain the slightly better overall results when combining WDC and Wiki data. Tab. 7 shows the official results on the hidden test set.

| | P | R | F1 |
|---|---|---|---|
| level 1 | 89.75 | 89.44 | 89.38 |
| level 2 | 88.66 | 88.22 | 88.05 |
| level 3 | 82.45 | 81.24 | 80.86 |
| Average | 86.96 | 86.30 | 86.10 |

Table 7: Results on the hidden test set (weighted averages).

## 5 Discussion and Improvements

We suggest a language model driven approach for identifying whether two texts describe the same product and which category they belong to. Using this text-only approach we left out additional available metadata, which, if successfully included, may allow for even better results. This simple approach nevertheless is enough to outperform baseline results, and while the models used might be complex, they can be easily set up and are a decent starting point for further research. For example, the integration of the whole product metadata for predictions like prices, brand or other features and some more in-depth error analyses to better generalize our models for unknown inputs would be promising experiments. The most important outcome and learning from the task was the observation that, even though we used pre-trained transformer models, more training data still significantly boosts performance and introduces valuable information to the classification process in both cases.

## 6 Acknowledgments

This research supported and funded in parts by the Development Bank of Saxony (SAB) under project number 100335729 and 100341518.

## References

1. Semantic Web Challenge ISWC2020 – Mining the Web of HTML-embedded Product Data, https://ir-ischool-uos.github.io/mwpd/index.html
2. Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., Guo, W.: *SEM 2013 shared task: Semantic textual similarity. In: Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity. pp. 32–43. Association for Computational Linguistics, Atlanta, Georgia, USA (Jun 2013), https://www.aclweb.org/anthology/S13-1004
3. Amoualian, H., Goswami, P., Ach, L., Das, P., Montalvo, P.: Sigir 2020 e-commerce workshop data challenge overview
4. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). pp. 1–14. Association for Computational Linguistics, Vancouver, Canada (Aug 2017). https://doi.org/10.18653/v1/S17-2001, https://www.aclweb.org/anthology/S17-2001
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs] (Oct 2018), http://arxiv.org/abs/1810.04805, arXiv: 1810.04805
6. Dolan, W.B., Brockett, C.: Automatically constructing a corpus of sentential paraphrases. In: Proceedings of the Third International Workshop on Paraphrasing (IWP2005) (2005), https://www.aclweb.org/anthology/I05-5002
7. Ganitkevitch, J., Van Durme, B., Callison-Burch, C.: PPDB: The paraphrase database. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 758–764. Association for Computational Linguistics, Atlanta, Georgia (Jun 2013), https://www.aclweb.org/anthology/N13-1092
8. GS1: Global Product Classification (GPC) - Standards (dec 2019), https://www.gs1.org/standards/gpc
9. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. pp. 427–431. Association for Computational Linguistics (April 2017)
10. Kim, Y.: Convolutional Neural Networks for Sentence Classification. arXiv:1408.5882 [cs] (Sep 2014), http://arxiv.org/abs/1408.5882, arXiv: 1408.5882
11. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs] (Jan 2017), http://arxiv.org/abs/1412.6980, arXiv: 1412.6980
12. Li, J., Dou, Z., Zhu, Y., Zuo, X., Wen, J.R.: Deep cross-platform product matching in e-commerce. Information Retrieval Journal **23**(2), 136–158 (2020)
13. Liu, J., Chang, W.C., Wu, Y., Yang, Y.: Deep Learning for Extreme Multi-label Text Classification. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 115–124. SIGIR '17, ACM, New York, NY, USA

(2017). https://doi.org/10.1145/3077136.3080834, http://doi.acm.org/10.1145/3077136.3080834, event-place: Shinjuku, Tokyo, Japan
14. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv:1907.11692 [cs] (Jul 2019), http://arxiv.org/abs/1907.11692, arXiv: 1907.11692
15. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. CoRR (Jan 2013), http://arxiv.org/abs/1301.3781, arXiv: 1301.3781
16. Mudgal, S., Li, H., Rekatsinas, T., Doan, A., Park, Y., Krishnan, G., Deep, R., Arcaute, E., Raghavendra, V.: Deep learning for entity matching: A design space exploration. In: Proceedings of the 2018 International Conference on Management of Data. p. 19–34. SIGMOD '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3183713.3196926, https://doi.org/10.1145/3183713.3196926
17. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
18. Primpeli, A., Peeters, R., Bizer, C.: The wdc training dataset and gold standard for large-scale product matching. In: Companion Proceedings of The 2019 World Wide Web Conference. p. 381–386. WWW '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3308560.3316609, https://doi.org/10.1145/3308560.3316609
19. Ristoski, P., Petrovski, P., Mika, P., Paulheim, H.: A machine learning approach for product matching and categorization. Semantic web **9**(5), 707–728 (2018)
20. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In: NeurIPS $EMC^2$ Workshop (2019)
21. Schuster, M., Nakajima, K.: Japanese and korean voice search. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 5149–5152. IEEE (2012)
22. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 1715–1725. Association for Computational Linguistics, Berlin, Germany (Aug 2016). https://doi.org/10.18653/v1/P16-1162, https://www.aclweb.org/anthology/P16-1162
23. Shah, K., Kopru, S., Ruvini, J.D.: Neural network based extreme classification and similarity models for product matching. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers). pp. 8–15 (2018)
24. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research **15**(1), 1929–1958 (2014)
25. Vrandečić, D., Krötzsch, M.: Wikidata: A free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (Sep 2014). https://doi.org/10.1145/2629489
26. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Brew, J.: Huggingface's transformers: State-of-the-art natural language processing. ArXiv **abs/1910.03771** (2019)
27. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al.: Google's neural machine translation

system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144v2 (2016)
28. Xiao, L., Huang, X., Chen, B., Jing, L.: Label-Specific Document Representation for Multi-Label Text Classification. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 466–475. Association for Computational Linguistics, Hong Kong, China (Nov 2019). https://doi.org/10.18653/v1/D19-1044, https://www.aclweb.org/anthology/D19-1044
29. Xu, W., Callison-Burch, C., Dolan, B.: SemEval-2015 task 1: Paraphrase and semantic similarity in twitter (PIT). In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015). pp. 1–11. Association for Computational Linguistics, Denver, Colorado (Jun 2015). https://doi.org/10.18653/v1/S15-2001, https://www.aclweb.org/anthology/S15-2001