Improving Hierarchical Product Classification using Domain-specific Language Modelling

Alexander Brinkmann
alex.brinkmann@informatik.uni-mannheim.de
Data and Web Science Group
University of Mannheim
Mannheim, Germany

Christian Bizer
chris@informatik.uni-mannheim.de
Data and Web Science Group
University of Mannheim
Mannheim, Germany

ABSTRACT

In order to deliver a coherent user experience, product aggregators such as market places or price portals integrate product offers from many web shops into a single product categorization hierarchy. Recently, transformer models have shown remarkable performance on various NLP tasks. These models are pre-trained on huge cross-domain text corpora using self-supervised learning and fine-tuned afterwards for specific downstream tasks. Research from other application domains indicates that additional self-supervised pre-training using domain-specific text corpora can further increase downstream performance without requiring additional task-specific training data. In this paper, we first show that transformers outperform a more traditional fastText-based classification technique on the task of assigning product offers from different web shops into a product hierarchy. Afterwards, we investigate whether it is possible to improve the performance of the transformer models by performing additional self-supervised pre-training using different corpora of product offers, which were extracted from the Common Crawl. Our experiments show that by using large numbers of related product offers for masked language modelling, it is possible to increase the performance of the transformer models by 1.22% in wF1 and 1.36% in hF1 reaching a performance of nearly 89% wF1.

CCS CONCEPTS

• **Information systems** → **Self-supervised Learning**; **Hierarchical Classification**; • **Computing methodologies** → **Neural Networks**.

KEYWORDS

e-commerce, hierarchical product classification, self-supervised learning, deep learning

ACM Reference Format:
Alexander Brinkmann and Christian Bizer. 2021. Improving Hierarchical Product Classification using Domain-specific Language Modelling. In *Proceedings of Workshop on Knowledge Management in e-Commerce @ The Web Conference '21 (The Web Conference '21).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

The Web Conference '21, April 16, 2021, Online

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Product aggregators like market places or price portals support customers in finding the right offer for their desired product. To ensure a good customer experience, product aggregators integrate heterogeneous product offers from large numbers of online shops into their own product categorization hierarchy. This hierarchical product classification task is a major challenge for product aggregators as most shops use their own proprietary categorization hierarchy as well as differing titles and descriptions for the same product. A promising technique to improve hierarchical product classification are pre-trained transformer models [19, 22]. These pre-trained transformer models have recently shown success for many NLP tasks [3, 5, 10, 12, 16, 20]. The training of transformer models involves two steps [3, 12]:

(1) The transformer model is pre-trained on a huge corpus of texts from books, news, online forums and stories using self-supervised Masked Language Modelling (MLM).
(2) The resulting model is fine-tuned for specific downstream tasks using task-specific training data.

During the first pre-training step the model acquires general knowledge on language representation. This knowledge is transferred to the downstream task. In related work the pre-training step is extended by additionally pre-training the transformer model on domain-specific text corpora [1, 11, 22]. In these works, the extended self-supervised pre-training results in improved performance on the downstream task.

Motivated by these findings, we investigate how an extended pre-training using heterogeneous product offers from the Web can improve model performance on the task of hierarchical product classification. For this purpose, we use product offers, which the Web Data Commons project[1] has extracted from the Common Crawl[2]. To identify offers and their properties, the project relies on schema.org[3] annotations in the HTML pages of the web shops [14]. The annotations enable the reliable extraction of the offer's title, the description of the offered product, as well as the offer's categorization within the proprietary categorization hierarchy of the specific web shop. The heterogeneous category values are of special interest, because the categories contain information about the product classification of the web shop. While being heterogeneous and web shop specific, previous work has show that this knowledge about product categories is beneficial for categorizing products into a single central product hierarchy [13, 23].

[1] http://webdatacommons.org/largescaleproductcorpus/v2/
[2] https://commoncrawl.org/
[3] https://schema.org/

We experiment with three different product corpora for pre-training that differ in size and relatedness to the downstream task. Through these different characteristics we measure the influence of size and relatedness of the pre-training corpus on the downstream hierarchical product classification task. Additionally, we experiment with different hierarchical classification methods. The methods combine RoBERTa$_{base}$ [12] with various classification heads in order to evaluate different approaches for exploiting the product hierarchy. We evaluate the classification methods using two product classification tasks involving product offers from many different web shops.

The contributions of this paper are as follows:

- We are the first to show that the performance of transformer models can be improved for the task of hierarchical product classification by performing additional pre-training using a corpus of related product offers.
- We show that using related product offers results in a better performance compared to randomly sampled product offers.

This paper is structured as follows: Section 2 introduces the classification models that will later be used for the experiments. Section 3 describes the evaluation tasks. While Section 4 presents the results of baseline experiments without additional language modelling. The effects of domain-specific MLM for hierarchical product classification are investigated in Section 5. Section 6 discusses related work. All data and code needed to replicate the results are available online[4].

2 CLASSIFICATION MODELS

This section introduces the classification models used for the experiments. The architecture of all classification models is composed of a pre-trained RoBERTa$_{base}$ transformer model and a task-specific classification head. RoBERTa$_{base}$ is chosen due to its recent success on related Natural Language Processing (NLP) tasks [12]. Figure 1 gives an overview of the complete procedure used for pre-training and fine-tuning the transformer models for hierarchical product classification. As we will describe in Section 5, the standard RoBERTa$_{base}$ model is pre-trained in Step 1 using a language modelling head and a corpus of product offers in order to inject additional domain-specific knowledge into the model. Afterwards in Step 2, the model resulting from Step 1 is fine-tuned for hierarchical product classification using a task-specific classification head.

For the classification the RoBERTa$_{base}$ model encodes the input text of the product offer. The first token of the encoded product offer is handed over to the classification head. This first token is referred to as [CLS] token. The [CLS] token serves as an encoded representation for the presented product offer. Based on the [CLS] token the classification head predicts the categories in the target product hierarchy for the presented product offer. One category is predicted for each level of the product hierarchy to evaluate the classification head's performance using evaluation metrics, which are specifically designed for hierarchical classification tasks. Section 4 will introduces the evaluation metrics in more detail. Since it can be assumed that the product hierarchy contains valuable information, the given product classification challenge is tackled with two hierarchical classification approaches. These hierarchical

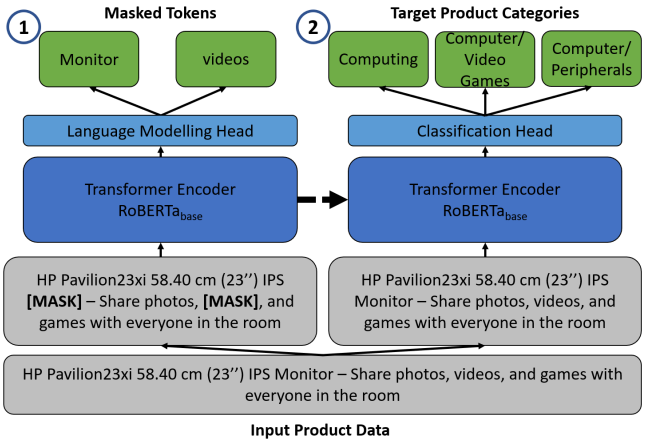

Figure 1: Overview of the (1) pre-training and (2) fine-tuning procedure, which combines a transformer model with a language modelling head first and afterwards with a task-specific classification head.

classification approaches exploit the product hierarchy. One approach is based on a hierarchical softmax classification. The other one is based on a Recurrent Neural Network (RNN). To measure the impact of the hierarchy exploitation, a flat classification approach without a explicit hierarchy usage is implemented, as well. All three approaches are explained in the following. Additionally, for one baseline model RoBERTa$_{base}$ is replaced by fastText[5], a state of the art neural network architecture for language representations [8]. This fastText model is combined with a flat classification head to set the classification results of the transformer models into a greater context.

2.1 Flat Classification

A general approach for many classification tasks using transformers is to add a linear layer on top of the transformer [1, 3, 12, 22]. This linear layer makes a prediction based on the [CLS] token. As this classification approach only assigns products to lowest level of categories, the parent categories are inferred using the product hierarchy. For training a cross-entropy loss is used.

2.2 Hierarchical Softmax

The hierarchical softmax classification head makes a prediction along all possible category paths from the root category to the leaf categories to obtain the probability that the presented product offer belongs to the given category path. To arrive at a probability for a category path, a local classifier is trained for each category in the product hierarchy. The local classifier is composed of a linear layer and a sigmoid activation function. The local classifier predicts the probability of a category to be part of the category path. The product of all local predictions along a category path is the probability of a category path to be predicted for the presented product offer. Using softmax the most probable category path among all category paths is chosen. The input of the local classifiers is the transformer's

[4]https://github.com/abrinkmann/productCategorization

[5]https://github.com/facebookresearch/fastText

[CLS] token. For training the cross-entropy loss is calculated per local classifier and per global category path. The combined loss deals with both the local impact of a single classifier and the global impact of a combination of classifiers along a category path.

2.3 Recurrent Neural Network

For the third classification head a RNN sequentially predicts a node for each level in the product hierarchy. The input for this classification head are the transformer's [CLS] token and a hidden state with the same size as the token. The hidden state is initialised with zeros. Inside the classification head both token representation and hidden state are concatenated. Based on the concatenated matrix a linear layer makes a prediction for the first level in the product hierarchy. A second linear layer updates the hidden state. The updated hidden state is fed back into the RNN to predict the next level in the product hierarchy. This procedure is repeated until a category is predicted for each level of the product hierarchy. During training the cross-entropy loss is calculated for each predicted category of the product hierarchy.

3 EVALUATION TASKS

This section introduces the hierarchical product classification tasks that are used for the evaluation. The objective of the tasks is to assign product offers from different web shops to the correct category in a single central product hierarchy.

3.1 MWPD Task

In the Mining the Web of Product Data (MWPD) challenge [6] the MWPD task [24] was used for benchmarking. The MWPD challenge was part of the International Semantic Web Conference (ISWC2020). In the product classification task of the MWPD challenge participants have to sort product offers from different web shops into the GS1 Global Product Classification standard (GPC)[7] [24]. GPC classifies product offers into a product hierarchy based on their essential properties and their relationship to other products. For the gold standard of the MWPD product classification data set the extracted product offers are manually assigned to the first three levels of the GPC.

3.2 Icecat/WDC222 Task

The training set of the Icecat/WDC222 task[8] is built based on the Open Icecat product data catalogue[9]. The Open Icecat product data catalog provides well maintained and normalized product information. For this work the attributes title, description, category and Global Trade Item Number (GTIN) are considered. All products belonging to the root category "Computers & Electronics" are extracted. For the classification the following three levels of the product hierarchy are considered. In order to evaluate whether a classifier is able to correctly classify heterogeneous product offers, the test set of the Icecat/WDC222 task consists of selected product offers from the Web Data Commons (WDC) Product Corpus[10]. This corpus contains offers from 79 thousand different websites, which use schema.org annotations. Using the GTIN the product offers are assigned to one out of 222 leaf categories in the Icecat product hierarchy. All assignments are manually verified. For all product offers the values of the attributes title, descriptions and GTIN are extracted. As the Icecat training set contains normalized product offers and the WDC222 test set contains heterogeneous product offers, the Icecat/WDC222 task measures the transferability of a classifier trained on clean product offers and transferred to a scenario involving heterogeneous product offers.

Table 1 shows that the MWPD training set is rather small compared to the training set of the Icecat use case, but the product offers of the MWPD task are drawn from a comparably large number of different hosts. The WDC222 test set is again rather small but covers more hosts than the Icecat training set. These high numbers of hosts are an indication for more heterogeneity, because the product offers are differently represented by different hosts. The analysis of the median and maximum number of records per category of both use cases as shown in Table 2 reveals that the distribution of product offers among the categories is skewed towards a small number of categories. This distribution is common for hierarchical classification tasks [17]. The missing description values of the Icecat/WDC222 task are a sign that the description might harm a classifier's performance if the classifier is trained on the Icecat training set and applied to the WDC222 test set.

4 BASELINE EXPERIMENTS

In order to set baselines, we apply the classification models described in Section 2 to both evaluation tasks that were introduced in Section 3. This section describes the setup as well as the results of the baseline experiments.

4.1 Evaluation Metrics

We use the average weighted F1 (wF1) score and the hierarchical F1 (hF1) score to evaluate the performance of the different models. Both scores are designed for hierarchical classification tasks [9, 24]. The wF1 score is calculated as proposed by the organisers of the MWPD challenge to compare our results to the results presented by the challenge participants. The calculation of the wF1 is shown in equation 1.

$$Average\ weighted\ F1\ (wF1) = \sum_{j=1}^{L} \frac{1}{L} \sum_{i=1}^{K_j} \frac{n_i}{N} F_i \qquad (1)$$

First, the F1 score of every category i in the hierarchy is calculated. To calculate the weighted F1 score per hierarchy level, the $F1_i$ score for each category i is weighted by number of true instances n_i for each category i divided by the total number of instances N across all categories K on a specific hierarchy level. For the average weighted F1 (wF1) score the weighted F1 scores per hierarchy level are divided by the total number of levels L in the hierarchy and summed up [24]. For the hF1 score all target and prediction categories of the different levels in the product hierarchy are considered to calculate the F1 score. This way the hF1 score is suitable for hierarchical classification tasks, as it directs higher credit to partially correct classifications, considers the distance of errors to the

[6] https://ir-ischool-uos.github.io/mwpd/
[7] https://www.gs1.org/standards/gpc
[8] http://data.dws.informatik.uni-mannheim.de/largescaleproductcorpus/categorization/
[9] https://icecat.biz/en/menu/channelpartners/index.html
[10] http://webdatacommons.org/largescaleproductcorpus/v2/

Table 1: Evaluation Task Statistics

Evaluation Task	No. Records Training Set	No. Records Test Set	No. Hosts Training Set	No. Hosts Test Set	No. Nodes in Hierarchy	Avg. Hierarchy Depth
MWPD	10,012	3,107	1,547	878	396	3
Icecat/WDC222	765,743	2,984	1	112	410	2.44

Table 2: Attribute Statistics

Evaluation Task	Data Set	Median No. Characters Title	Missing Values Description	Median No. Characters Description	Median No. Records per Category	Maximum No. Records per Category
MWPD	Train	50	0%	304	7	3,228
MWPD	Test	48	0%	365	4	799
Icecat/WDC222	Train	57	29.65%	1,099	215	145,020
Icecat/WDC222	Test	54	22.72%	140.5	3	516

correct category and errors higher up in the hierarchy are punished more severely [9]. Additionally, McNemar's test is applied to verify significantly different model performances on the test set [4]. For McNemar's test it is determined if a classifier's prediction is correct or incorrect first. Afterwards, the numbers of correctly predicted product offers by the first classifier and incorrectly predicted product offers by the second classifier (correct/ incorrect) and vice versa (incorrect/ correct) are calculated. Using these numbers of correct/ incorrect and incorrect/ correct predictions as well as a significance level of 0.01, McNemar's test determines if the proportion of errors and consequently the performance of the two compared classifiers on the test set is significantly different.

4.2 Experimental Setup

We use the following hyperparameter setting for the experiments: The learning rate is set to 3e-5 for the Icecat/WDC222 task and to 5e-5 for the MWPD task. We use a batch size of 8 and a linear weight decay of 0.01. All fine-tuning experiments are run for 25 epochs on the MWPD data set and 10 epochs on the Icecat/WDC222 data set. The different learning rates and numbers of epochs are a result of multiple experiment runs. In this setting the average results on the test set over three randomly initialized runs are reported for every experiment. For McNemar's test a majority voting among the results of the different runs is performed. As input for the classification models the values of the attributes title and description are lowercased and excessive white-spaces are removed. For the experiments in this section a $\text{RoBERTa}_{\text{base}}$ model is used to obtain a product representation, which is consumed by different classification heads to obtain a classification.

4.3 Results

Table 3 and Table 4 show the results of the baseline experiments. We use the following naming convention in order to refer to the different models: <input attributes>-<transformer model>-<classification head>. If the values of <input attribute> is 1, only the title is used as input. If the values of <input attribute> is 2, both title and description are used as input. In this section <transformer model> is either base for $\text{RoBERTa}_{\text{base}}$ or fast for fastText. The value of <classification head> refers to one of the classification heads introduced in Section 2 flat, hierarchical or rnn. Experiments with the same capital letter in the column "Same Error Rate - McNemar's Test" share the same error proportion on the test set according to McNemar's significance test. Otherwise the experiment's error proportion is significantly different.

The experimental results for the MWPD task are shown in Table 3. Setting the results of the model 2-fast-flat into context to the other models shows that all transformer-based approaches outperform the fastText baseline model. A comparison of the models 1-base-flat and 2-base-flat reveals that adding the description as input improves the performance of the classification. The comparison of the three 2-base-flat, 2-base-hierarchical and 2-base-rnn indicates that exploiting the product hierarchy does not provide a major benefit for the MWPD task. The performance difference of the models 2-base-flat and 2-base-rnn is not significant and 2-base-hierarchical performs worse than the other two models. Thus, 2-base-flat is chosen as baseline model for the experiments with domain-specific language modelling that will be described in Section 5.

Table 4 shows the results of the different models for the Icecat/WDC222 task. Again, the fastText based model 1-fast-flat is outperformed by the transformer-based models. The comparison of the models 1-base-flat and 2-base-flat shows that adding the description harms the performance of the trained classifier. This finding is expected given the high percentage of missing values and the difference in the median number of characters between training and test set as shown in Table 2. This comparison of the models 1-base-flat and 1-base-rnn shows that the RNN leads to a performance gain. A reason for this improvement might be the huge size of the Icecat training data set compared to the size of the MWPD data set. This size enables the RNN classification head to better learn the encoded hierarchy of the labels, which is beneficial for the classification on the test data set.

Table 3: Experimental results without Language Modelling - MWPD Task

Model	Attributes	Classification Head	wF1	Δ wF1	hF1	Δ hF1	Same Error Rate McNemar's
2-fast-flat	Title, Description	Flat	84.26		82.68		
1-base-flat	Title	Flat	87.01	2.75	87.03	4.35	
2-base-flat	Title, Description	Flat	87.52	3.26	87.62	4.94	A
2-base-hierarchical	Title, Description	Hierarchical	87.00	2.74	87.47	4.79	
2-base-rnn	Title, Description	RNN	87.47	3.21	87.67	4.99	A

Table 4: Experimental results without Language Modelling - Icecat/WDC222 Task

Model	Attributes	Classification Head	wF1	Δ wF1	hF1	Δ hF1	Same Error Rate McNemar's Test
1-fast-flat	Title	Flat	77.58		83.64		
1-base-flat	Title	Flat	83.36	5.78	84.69	1.05	
2-base-flat	Title, Description	Flat	80.91	3.33	81.48	-2.16	
1-base-rnn	Title	RNN	86.56	8.98	85.61	1.97	

5 DOMAIN-SPECIFIC LANGUAGE MODELLING

After establishing baseline results in the previous section, we now investigate the effect of domain-specific language modelling on the performance of the $\text{RoBERTa}_{\text{base}}$ models for hierarchical product classification. In this section the extraction of the domain-specific product offer corpora and the applied MLM approach are explained. The effects of domain-specific MLM on the domain-specific corpora are demonstrated by fine-tuning the newly pre-trained transformer models on the MWPD use case. The results of this fine-tuning are set into relation to the baseline results without domain-specific pre-training.

5.1 Product Corpora

In total three different product corpora are used for additional domain-specific MLM. All three product corpora contain product offers extracted from the WDC Product Corpus[11]. The WDC Product Corpus contains structured data for 365,577,281 product offers that are extracted from 581,482 different hosts. The structured data about product offers is extracted using schema.org annotations. Schema.org annotations are used by the web shops to enrich the search results of product aggregators with the web shop's product offers [14]. To retrieve product offers from the WDC Product Corpus three strategies are applied. For the first two domain-specific product corpora 1,547 top-level domains of product offers contained in the MWPD training set are identified. Using these top-level domains, product offers from the same top-level domains are extracted from the WDC Product Corpus. This heuristic assumes that all products offered by a single shop (top-level domain) are related. The first product corpus contains 75,248 product offers and will be called in the following Small Related product corpus. The second product corpus contains 1,185,884 product offers and is referred to as Large Related product corpus. Through these two corpora the effect of the number of the product offer on MLM is measured. For the third corpus a large random sample of product offers is extracted from the WDC product corpus. This corpus is referred to as Large Random product corpus. The Large Random product corpus enables us to measure the effect of relatedness of product offers on domain-specific language modelling.

All extracted product offers have a title and at least one of the attributes description or category. The attributes are identified using these schema.org annotations:

- Title: Product/name
- Description: Product/description
- Category: Product/category, Product/breadcrumb and BreadcrumbList

The attribute category is associated with multiple annotations, because all of these annotations categorise a product. Breadcrumbs are used by online shops to help users navigate the product portfolio. In this sense a list of breadcrumbs reflects categories of different levels in a product hierarchy. By combining the different annotations the percentage of missing category values is decreased. In a last pre-processing step all attribute values of the final product corpora are lowercased and excessive white spaces are removed.

Table 5 gives an overview of the product corpora's characteristics. For the two related product corpora the median number of records per hosts is higher compared to the Large Random product corpus. This shows the focus of the related corpora on the top-level domains extracted from the training set. The Large Random product corpus does not have this focus. Consequently, the number of hosts is a lot higher and the median number of records per host is lower. Table 5 shows that the product corpus Large Random has a comparably high percentage of missing description values. The Large Related product corpus has a lot of missing category values. These characteristics might influence the outcome of MLM.

[11] http://webdatacommons.org/structureddata/2017-12/stats/schema_org_subsets.html

Table 5: Product Corpora for Language Modelling

Product Corpus	Size	No. Hosts	Median No. Records per Host	Max No. Records per Host	Median No. Characters Title	Median No. Characters Description	Missing Values Description	Median No. Characters Category	Missing Values Category
Small Related	75,248	1,160	100	400	38	310	10.43%	24	29.69%
Large Related	1,185,884	1,505	48	5,885	41	275	7.06%	22	68.90%
Large Random	1,029,063	98,421	2	2,878	34	39	72.99%	70	44.32%

5.2 Attribute Combinations

For MLM the attributes title, category and description are used. The product categories do not follow the categories of the downstream hierarchical product classification, because these categories assigned by the online shops themselves. Still these categories can contain valuable product information. In the basic setup the attribute values are concatenated to a single line text representation of the product. This default attribute combination is referred to as Title-Cat-Desc. To measure the effect of the heterogeneous categories, two additional product text representations are used for MLM. In one scenario only title and description are considered for MLM. The categories are disregarded. This scenario is referred to as Title-Desc and encoded in the model's name as <transformer model>$_{\text{nocat}}$. As alternative setup, the product attributes are split into two lines. One line contains the attribute values of title and category and the other line contains the attribute values of title and description. This scenario is referred to as Title-Cat/Title-Desc and encoded in the model's name as <transformer model>$_{\text{ext}}$. This way the influence of the heterogeneous categories during language modelling can be measured. Due to the smaller size and the low percentage of missing category values of the product corpus Small Related as shown in Table 5, the impact of using the category information on the model performance is evaluated using this product corpus.

5.3 MLM Procedure

The pre-training used to inject knowledge about product offers into the RoBERTa $_{\text{base}}$ model follows the MLM procedure used to pre-train RoBERTa $_{\text{base}}$ initially. During MLM in each epoch a random sample of tokens from the input sequence is selected and replaced by the special token [MASK]. Uniformly 15% of the input tokens are selected for possible replacement. Of these selected tokens, 80% are replaced with [MASK], 10% are left unchanged and 10% are replaced by a randomly selected vocabulary token. For MLM a language modelling head predicts the masked tokens of the input. The MLM objective is a cross-entropy loss on predicting the masked tokens [3, 12]. For the downstream hierarchical product classification the language modelling head is replaced by one of the task-specific classification heads introduced in Section 2.

5.4 Experimental Setup

For pre-training the RoBERTa$_{\text{base}}$ models on the different product corpora, the chosen hyperparameters are a batch size of 4, a learning rate of 5e-5 and a linear weight decay of 0.01. All models are pre-trained for 5 epochs. The downstream hierarchical product classification follows the same settings as the baseline experiments described in Section 4. In this setting the average results on the test set over three randomly initialized runs for each experimental setup are reported. Based on their usefulness for the baseline models both attributes title and description are used as input for hierarchical product classification on the MWPD task. For the Icecat/WDC222 task only the title is used as input. Since the collection of the product corpora focuses on the MWPD task, the conducted experiments with an extended domain-specific MLM focus on the MWPD task, too. The best performing pre-trained model on the MWPD task is transferred to the Icecat/WDC222 task.

Table 6 and Table 7 show the experimental results of an extended domain-specific MLM for hierarchical product classification. To reference the different models the same encoding as in Section 4 is used. For <transformer model> the identifiers rel_small for pre-training on the Small Large corpus, rel_large for pre-training on the Related Large corpus and rand_large for pre-training on the Random Large corpus are added. Experiments with the same capital letter in the column "Same Error Rate - McNemar's Test" share the same error proportion on the test set according to McNemar's significance test. Otherwise the experiment's error proportion is significantly different.

5.5 Effect of Using Different Product Corpora

Table 6 shows that the baseline model 2-base-flat is outperformed by all other models on the MWPD task. This demonstrates the positive impact of domain-specific MLM on hierarchical product classification. The performance increase of the model 2-rel_large-rnn compared to the models 2-rel_small-rnn and 2-rand_large indicates that a large number of related product offers has the most positive effect on the model's performance. Among the different classification heads, the results of the models 2-rel_large-flat, 2-rel_large-hierarchical and 2-rel_large-rnn show that the RNN profits most from domain-specific pre-training. In general, our best model 2-rel_large-rnn outperforms the baseline model 2-base-flat by 1.22 wF1 and 1.18 hF1 points on the MWPD task. According to the result of McNemar's test this performance difference is significant.

Table 7 reveals that the models 1-rel_large-rnn and 1-base-rnn have the same performance on the Icecat/WDC222 task. This underlines that the pre-training corpus has to be as similar as possible to the hierarchical product classification task to gain a significant performance boost from pre-training.

5.6 Effect of Using Web Shop Categories

The results in Table 6 indicate a slightly positive effect of using the heterogeneous categorization information from the original

Table 6: Experimental results with Language Modelling - MWPD Task

Model	Product Corpus MLM	Attribute Combination MLM	Classification Head	wF1	Δ wF1	hF1	Δ hF1	Same Error Rate McNemar's Test
2-base-flat	None	None	Flat	87.52		87.62	4.94	B
2-rel_large-flat	Large Related	Title-Cat-Desc	Flat	87.61	0.09	87.70	0.08	B
2-rel_small-rnn	Small Related	Title-Cat-Desc	RNN	88.31	0.79	88.47	0.85	C
2-rel_large-rnn	Large Related	Title-Cat-Desc	RNN	88.74	1.22	88.80	1.18	
2-rand_large-rnn	Large Random	Title-Cat-Desc	RNN	88.19	0.67	88.34	0.72	
2-rel_large-hierarchical	Large Related	Title-Cat-Desc	Hierarchical	88.44	0.92	88.60	0.98	
2-rel_small$_{\text{nocat}}$-rnn	Small Related	Title-Desc	RNN	88.27	0.75	88.41	0.79	C
2-rel_small$_{\text{ext}}$-rnn	Small Related	Title-Cat/ Title-Desc	RNN	88.74	1.22	88.98	1.36	

Table 7: Experimental results with Language Modelling - Icecat/WDC222 Task

Model	Product Corpus MLM	Attribute Combination MLM	Classification Head	wF1	Δ wF1	hF1	Δ hF1	Same Error Rate McNemar's Test
1-base-rnn	None	None	Flat	86.56		85.58		D
1-rel_large-rnn	Large Related	Title-Cat-Desc	RNN	86.38	-0.18	85.61	+0.03	D

web shops as additional feature during pre-training. A comparison of the models 2-rel_small$_{\text{nocat}}$-rnn and 2-rel_small-rnn shows that disregarding the web shop categories has a negative but not significant impact on the model's performance. In the extended scenario of model 2-rel_small$_{\text{ext}}$-rnn, the performance improves up to the performance level of the model 2-rel_large-rnn. The model 2-rel_small$_{\text{ext}}$-rnn significantly outperforms the baseline model 2-base-flat by 1.22 wF1 and 1.36 hF1 points on the MWPD task. A reason for these results might be that doubling the product representations during pre-training has a positive impact, because it almost doubles the amount of input text available for pre-training. This effect is comparable to doubling the number of training epochs, which might improve the performance results, too. Another reason might be the length of the different attributes. The values of the attributes title and category are rather short compared to the attribute values of the description as shown by the median number of characters in Table 2. If the product offer is represented by a single line, the generated masked tokens during MLM are more likely part of the description than part of title or category. If mainly tokens from the description are masked, the model learns to better represent these long descriptions. At the same time, it can be assumed that the title is more informative than the lengthy description. By presenting the product offers twice with different attribute combinations to the model during pre-training the disturbing effect of long descriptions is reduced. This allows the model to better exploit the heterogeneous categories and the title. From our results we can conclude that the heterogeneous categories from the web shops have a slightly positive but not significant impact on the model's performance.

6 RELATED WORK

This section discusses related work on the domain adaptation of transformer models and gives an overview of the state of the art on hierarchical product classification using transformer models.

6.1 Domain Adaptation

Recently, the technique of pre-training large transformer models in a text rich environment using self-supervised learning and fine-tuning them on a downstream task has become successful in NLP [3, 12, 15, 20]. BERT is one of these transformer models and was pre-trained on publicly available text corpora consisting such as the text of books and the English Wikipedia [3]. Through pre-training the model obtains the ability to encode natural language [7, 15, 16]. This knowledge about natural language is then transferred to downstream tasks. Pre-trained domain-specific models like SCIBERT, BioBERT and E-BERT, which are based on BERT, have shown that additional pre-training on domain-specific text improves the performance on downstream domain-specific tasks [1, 11, 22]. E-BERT for example uses adaptive masking on a product and a review corpus during pre-training to learn e-Commerce knowledge on phrase-level and on product-level. Pre-training enables E-BERT to outperform a BERT based model on different downstream tasks related to e-commerce [22]. Comparing the effects of adaptive masking and random masking using the product offer corpora that were created for this paper is an interesting direction for future work.

6.2 Hierarchical Product Classification

Related work shows that successfully exploiting this hierarchical structure can improve classification results [6, 17–19, 21]. Team Rhinobird [19], the winners of the MWPD challenge [24], combine the pre-trained transformer model BERT with a hierarchical classification head. The classification head sequentially predicts the categories of different levels in the product hierarchy. This is achieved by actively restricting the classes, which can be predicted on the lower levels based on the predicted parent level node. Team Rhinobird calls this approach Dynamic Masked Softmax. Through different BERT based representations an ensemble of classifiers with a Dynamic Masked Softmax head enables Rhinobird to reach a wF1 score of 88.08 on the MWPD task that is used in this paper.

Rhinobird [19] applies pseudo labelling on the unlabeled test data to further improve the performance of their model. Afterwards, their ensemble of models is partly trained using these pseudo labels. This procedure might leak information about the test set into the training process. Thus, we compare our models to the Rhinobird results without pseudo labelling. Our best model based on a domain-specifically pre-trained RoBERTa model and a RNN classification head achieves a performance of 88.74 wF1 points. This is an improvement of +0.66 points over Rhinobird's results. Given that Rhinobird achieves this good performance using an ensemble of models, future work could examine how an ensemble consisting of differently pre-trained and differently fine-tuned transformers can further improve the performance of our model. Team ASVinSpace uses a CNN based approach for language modelling with a multi-output classification head that predicts the categories of the different levels in the product hierarchy [2]. Our best model outperforms ASVinSpace's approach by +2.14 wF1 points.

7 CONCLUSION

This paper has shown that the performance of transformer models on the task of hierarchical product classification can be improved by performing additional self-supervised pre-training using a corpus of related product offers. Our experiments with three different domain-specific corpora of product offers demonstrate that a large corpus of related product offers leads to the highest performance gain if the attributes are concatenate to get a one-line product representation. If the product representation is split into two lines during pre-training, the result on the hierarchical product classification task is further improved even though only a small corpus of related products is used. With this approach and a task-specific classification head our best model outperforms the baseline model by 1.22 wF1 points and 1.36 hF1 points. The baseline model relies on general pre-training and a task-specific fine-tuning. The product representation is split into two lines such that the first line contains title and description and the second line contains title and category. This split enables the transformer to better exploit the product representations. Future research can examine how MLM can further exploit the inherent knowledge of the presented product offers to improve its efficiency during pre-training.

REFERENCES

[1] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. *arXiv:1903.10676 [cs]* (Sept. 2019). http://arxiv.org/abs/1903.10676 arXiv: 1903.10676.

[2] Janos Borst, Erik Krner, and Andreas Niekler. 2020. Language Model CNN-driven similarity matching and classification for HTML-embedded Product Data. In *CEUR Workshop Proceedings*, Vol. 2720. RWTH, Aachen, 11.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]* (May 2019). http://arxiv.org/abs/1810.04805 arXiv: 1810.04805.

[4] Thomas G. Dietterich. 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation* 10, 7 (Oct. 1998), 1895–1923. https://doi.org/10.1162/089976698300017197

[5] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. *arXiv:1905.03197 [cs]* (Oct. 2019). http://arxiv.org/abs/1905.03197 arXiv: 1905.03197.

[6] Dehong Gao, Wenjing Yang, Huiling Zhou, Yi Wei, Yi Hu, and Hao Wang. 2020. Deep Hierarchical Classification for Category Prediction in E-commerce System. In *3rd Workshop on e-Commerce and NLP (ECNLP 3)*. Online, 64–68. https://www.aclweb.org/anthology/2020.ecnlp-1.10.pdf

[7] Joel Hestness, Sharan Narang, Newsha Ardalani, Gregory Diamos, Heewoo Jun, Hassan Kianinejad, Md Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou. 2017. Deep Learning Scaling is Predictable, Empirically. *arXiv:1712.00409 [cs, stat]* (Dec. 2017). http://arxiv.org/abs/1712.00409 arXiv: 1712.00409.

[8] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. *arXiv:1607.01759 [cs]* (Aug. 2016). http://arxiv.org/abs/1607.01759 arXiv: 1607.01759.

[9] Svetlana Kiritchenko, Stan Matwin, and A Fazel Famili. 2005. Functional Annotation of Genes Using Hierarchical Text Categorization. In *BioLINK SIG: Linking Literature, Information and Knowledge for Biolog*. Detroit, Michigan, USA. https://www.researchgate.net/profile/Svetlana_Kiritchenko/publication/44046343_Functional_Annotation_of_Genes_Using_Hierarchical_Text_Categorization/links/09e4150b3962ea0f9c000000/Functional-Annotation-of-Genes-Using-Hierarchical-Text-Categorization.pdf

[10] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. https://openreview.net/forum?id=H1eA7AEtvS

[11] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *arXiv:1901.08746 [cs]* (Oct. 2019). https://doi.org/10.1093/bioinformatics/btz682 arXiv: 1901.08746.

[12] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]* (July 2019). http://arxiv.org/abs/1907.11692 arXiv: 1907.11692.

[13] Robert Meusel, Anna Primpeli, Christian Meilicke, Heiko Paulheim, and Christian Bizer. 2015. Exploiting Microdata Annotations to Consistently Categorize Product Offers at Web Scale. In *E-Commerce and Web Technologies (Lecture Notes in Business Information Processing)*, Heiner Stuckenschmidt and Dietmar Jannach (Eds.). Springer International Publishing, Cham, 83–99. https://doi.org/10.1007/978-3-319-27729-5_7

[14] Anna Primpeli, Ralph Peeters, and Christian Bizer. 2019. The WDC Training Dataset and Gold Standard for Large-Scale Product Matching. In *Companion Proceedings of The 2019 World Wide Web Conference (WWW '19)*. Association for Computing Machinery, New York, NY, USA, 381–386. https://doi.org/10.1145/3308560.3316609

[15] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. (2019), 24.

[16] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv:1910.10683 [cs, stat]* (July 2020). http://arxiv.org/abs/1910.10683 arXiv: 1910.10683.

[17] Carlos N. Silla and Alex A. Freitas. 2011. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery* 22, 1-2 (Jan. 2011), 31–72. https://doi.org/10.1007/s10618-010-0175-9

[18] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. 2018. Hierarchical Multi-Label Classification Networks. In *International Conference on Machine Learning*. PMLR, Stockholm, Sweden, 5075–5084. http://proceedings.mlr.press/v80/wehrmann18a.html ISSN: 2640-3498.

[19] L. Yang, E. Shijia, Shiyao Xu, and Yang Xiang. 2020. Bert with Dynamic Masked Softmax and Pseudo Labeling for Hierarchical Product Classification. In *CEUR Workshop Proceedings*, Vol. 2720. RWTH, Aachen. http://ceur-ws.org/Vol-2720/paper6.pdf

[20] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2020. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *arXiv:1906.08237 [cs]* (Jan. 2020). http://arxiv.org/abs/1906.08237 arXiv: 1906.08237.

[21] Ronghui You, Zihan Zhang, Ziye Wang, Suyang Dai, Hiroshi Mamitsuka, and Shanfeng Zhu. 2019. AttentionXML: Label Tree-based Attention-Aware Deep Model for High-Performance Extreme Multi-Label Text Classification. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc., 11. https://proceedings.neurips.cc/paper/2019/file/9e6a921fbc428b5638b3986e365d4f21-Paper.pdf

[22] Denghui Zhang, Zixuan Yuan, Yanchi Liu, Zuohui Fu, Fuzhen Zhuang, Pengyang Wang, Haifeng Chen, and Hui Xiong. 2020. E-BERT: A Phrase and Product Knowledge Enhanced Language Model for E-commerce. *arXiv:2009.02835 [cs]* (Sept. 2020). http://arxiv.org/abs/2009.02835 arXiv: 2009.02835.

[23] Shuo Zhang and Krisztian Balog. 2019. Auto-completion for Data Cells in Relational Tables. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, Beijing China, 761–770. https://doi.org/10.1145/3357384.3357932

[24] Ziqi Zhang, Christian Bizer, Ralph Peeters, and Anna Primpeli. 2020. MWPD2020: Semantic Web challenge on Mining the Web of HTML-embedded product data. In *CEUR Workshop Proceedings*, Ziqi Zhang (Ed.), Vol. 2720. RWTH, Aachen, 2–18. http://ceur-ws.org/Vol-2720/paper1.pdf ISSN: 1613-0073.