# Improving Hierarchical Product Classification using Domain-specific Language Modelling

Alexander Brinkmann, Christian Bizer
University of Mannheim
{alex.brinkmann, chris}@informatik.uni-mannheim.de

**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.*

## A 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 diverse titles and descriptions for the same product. A promising technique to improve hierarchical product classification are pre-trained transformer models [5, 14]. These pre-trained transformer models have recently shown success for many NLP tasks [2, 3, 12, 13, 19, 22]. The training of transformer models involves two steps [2, 3]:

1. Pre-Training: The 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. Fine-Tuning: The pre-trained model is fine-tuned for downstream tasks using task-specific training data.


**Bulletin of the IEEE Computer Society Technical Committee on Data Engineering**

During pre-training the model acquires general knowledge on language representation. This knowledge can be applied to solve down-stream tasks. In related work the pre-training step is extended by additionally pre-training the transformer model on domain-specific text corpora [4, 5, 7, 23]. In these works, the extended self-supervised pre-training results in improved performance on downstream tasks.

Motivated by these findings, we investigate whether additional pre-training using heterogeneous product offers from the Web can improve hierarchical product classification. For this purpose, we use product offers, which the Web Data Commons project[1] has extracted from the Common Crawl[2]. For identifying product offers and their attributes, the project relies on schema.org[3] annotations in the HTML pages of the web shops [20]. 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 [9, 17].

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 $\text{RoBERTa}_{\text{base}}$ [3] 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 B introduces the classification models that will later be used for the experiments. Section C describes the evaluation tasks. While Section D presents the results of baseline experiments without additional language modelling. The effects of domain-specific MLM for hierarchical product classification are investigated in Section E. Section F discusses related work. All data and code needed to replicate the results are available online[4].

# B Classification models

The architecture of all classification models is composed of a pre-trained $\text{RoBERTa}_{\text{base}}$ transformer model and a task-specific classification head. $\text{RoBERTa}_{\text{base}}$ is chosen due to its recent success on related Natural Language Processing (NLP) tasks [3]. Additionally, for one baseline model $\text{RoBERTa}_{\text{base}}$ is replaced by fastText[5], a state of the art neural network architecture for language representations [1]. For the classification the $\text{RoBERTa}_{\text{base}}$ model encodes the input text of the product offer. The first token of the encoded product offer is handed over to the classification head. Based on the first token, also referred to as [CLS] token, the classification head predicts a category for each product hierarchy level. Since it can be assumed that the product hierarchy contains valuable information, the given product classification challenge is tackled with three different classification heads. These

[1] http://webdatacommons.org/largescaleproductcorpus/v2/
[2] https://commoncrawl.org/
[3] https://schema.org/
[4] https://github.com/wbsg-uni-mannheim/productCategorization
[5] https://github.com/facebookresearch/fastText

classification heads try to exploit the upfront known product hierarchy. The three approaches are referred to as flat classification, hierarchical softmax and Recurrent Neural Network (RNN).

### B.1 Flat Classification

For the flat classification, a linear layer makes a prediction based on the [CLS] token. As this classification approach only assigns product offers to lowest level of categories, the parent categories are inferred using the product hierarchy. For training a cross-entropy loss is used.

### B.2 Hierarchical Softmax

The hierarchical softmax classification head predicts the category path with the highest probability for a product offer. To arrive at a probability for a category path, one local classifier is trained for each category in the product hierarchy. 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 a 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 [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.

### B.3 Recurrent Neural Network

For the third classification head a RNN sequentially predicts a category 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 [CLS] token. Based [CLS] token and hidden state a linear layer predicts the category for the current product hierarchy level. 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 in the product hierarchy. During training the cross-entropy loss is calculated for each predicted category.

## C 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 categories in a single central product hierarchy.

### C.1 MWPD Task

The MWPD task was used at the Mining the Web of Product Data (MWPD) challenge[6] [11] 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] [11]. 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.

[6] https://ir-ischool-uos.github.io/mwpd/
[7] https://www.gs1.org/standards/gpc

## C.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. For the classification the first 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 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 [8]. 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.

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

Table 1: Evaluation Task 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 |

Table 2: Attribute Statistics

[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/

# D Baseline Experiments

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

## D.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 [10, 11]. The wF1 score is calculated as proposed by the organisers of the MWPD challenge and shown in equation 4 [11].

$$Average\ weighted\ F1\ (wF1) = \sum_{j=1}^{L} \frac{1}{L} \sum_{i=1}^{K_j} \frac{n_i}{N} F_i \tag{4}$$

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 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 correct category and errors higher up in the hierarchy are punished more severely [10]. Additionally, McNemar's significance test is applied to verify significantly different model performances on the test set [21]. For the test it is determined if a classifier's prediction is correct or incorrect first. Second, 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.

## D.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.

## D.3 Results

The naming convention <input attributes>-<transformer model>-<head> is used to refer to the different models. If the value of <input attribute> is "1", only the title is used as input. If the value 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 <head> refers to one of the classification heads introduced in Section B "flat", "hier" for hierarchical or "rnn". Experiments with the same capital letter in the column "Same Error Rate" share the same error proportion on the test set according to the 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 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 as described in Section E. 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.

| Model | Attributes | Classification Head | wF1 | $\Delta$ wF1 | hF1 | $\Delta$ hF1 | Same Error Rate |
|---|---|---|---|---|---|---|---|
| 2-fast-flat | Title, Desc. | Flat | 84.26 | | 82.68 | | |
| 1-base-flat | Title | Flat | 87.01 | 2.75 | 87.03 | 4.35 | |
| 2-base-flat | Title, Desc. | Flat | 87.52 | 3.26 | 87.62 | 4.94 | A |
| 2-base-hier | Title, Desc. | Hierarchical | 87.00 | 2.74 | 87.47 | 4.79 | |
| 2-base-rnn | Title, Desc. | RNN | 87.47 | 3.21 | 87.67 | 4.99 | A |

Table 3: Experimental results without Language Modelling - MWPD Task

| Model | Attributes | Classification Head | wF1 | $\Delta$ wF1 | hF1 | $\Delta$ hF1 | Same Error Rate |
|---|---|---|---|---|---|---|---|
| 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, Desc. | Flat | 80.91 | 3.33 | 81.48 | -2.16 | |
| 1-base-rnn | Title | RNN | 86.56 | 8.98 | 85.61 | 1.97 | |

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

# E 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 $\text{RoBERTa}_{\text{base}}$ 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 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.

### E.1 Product Corpora

In total three different product corpora are used for 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 using schema.org annotations. The hosts use schema.org annotations to enrich the search results of product aggregators with their web shop's product offers [20]. Three strategies are applied to retrieve product offers from the WDC Product Corpus. For the first two domain-specific product corpora 1,547 top-level domains of product offers from 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 is called 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. 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.

| Product Corpus | No. Records | No. Hosts | Median No. Records per Host | Max No. Records per Host |
|---|---|---|---|---|
| Small Related | 75,248 | 1,160 | 100 | 400 |
| Large Related | 1,185,884 | 1,505 | 48 | 5,885 |
| Large Random | 1,029,063 | 98,421 | 2 | 2,878 |

Table 5: Size of Product Corpora

All extracted product offers have a title and at least one of the attributes description or category. The attributes are identified using the schema.org product annotations[12] name for title, description for description as well as category, breadcrumb and breadcrumbList for category. The attribute category is associated with multiple annotations, because different hosts use various annotations to categorise their products. Lastly, all attribute values are lowercased and excessive white spaces are removed. Table 6 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.

| Product Corpus | Median No. Characters Title | Median No. Characters Description | Missing Values Desccription | Median No. Characters Category | Missing Values Category |
|---|---|---|---|---|---|
| Small Related | 38 | 310 | 10.43% | 24 | 29.69% |
| Large Related | 41 | 275 | 7.06% | 22 | 68.90% |
| Large Random | 34 | 39 | 72.99% | 70 | 44.32% |

Table 6: Distribution of Attributes in the Product Corpora

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

## E.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.

## E.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 [2, 3]. For the downstream hierarchical product classification the language modelling head is replaced by one of the task-specific classification heads introduced in Section B.

## E.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 D. 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 **??** and Table **??** 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 D is used. For <transformer model> the identifiers rel_s for pre-training on the Small Large corpus, rel_l for pre-training on the Related Large corpus and rand_l for pre-training on the Random Large corpus are added. Experiments with the same capital letter in the column "Same Error Rate" share the same error proportion on the test set according to McNemar's significance test. Otherwise the experiment's error proportion is significantly different.

| Model | Product Corpus MLM | Attribute Combination MLM | Head | wF1 | Δ wF1 | hF1 | Δ hF1 | Same Error Rate |
|---|---|---|---|---|---|---|---|---|
| 2-base-flat | None | None | Flat | 87.52 | | 87.62 | 4.94 | B |
| 2-rel_l-flat | Large Related | Title-Cat-Desc | Flat | 87.61 | 0.09 | 87.70 | 0.08 | B |
| 2-rel_s-rnn | Small Related | Title-Cat-Desc | RNN | 88.31 | 0.79 | 88.47 | 0.85 | C |
| 2-rel_l-rnn | Large Related | Title-Cat-Desc | RNN | 88.74 | 1.22 | 88.80 | 1.18 | |
| 2-rand_l-rnn | Large Random | Title-Cat-Desc | RNN | 88.19 | 0.67 | 88.34 | 0.72 | |
| 2-rel_l-hierar. | Large Related | Title-Cat-Desc | Hierar. | 88.44 | 0.92 | 88.60 | 0.98 | |
| 2-rel_$s_{nocat}$-rnn | Small Related | Title-Desc | RNN | 88.27 | 0.75 | 88.41 | 0.79 | C |
| 2-rel_$s_{ext}$-rnn | Small Related | Title-Cat/ Title-Desc | RNN | 88.74 | 1.22 | 88.98 | 1.36 | |

Table 7: Experimental results with Language Modelling - MWPD Task

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

Table 8: Experimental results with Language Modelling - Icecat/WDC222 Task

### E.5 Effect of Using Different Product Corpora

Table **??** shows that the baseline model 2-base-flat is outperformed by all other models on the MWPD task. Our best model 2-rel_l-rnn outperforms the baseline model 2-base-flat by 1.22 wF1 and 1.18 hF1 points. According to the significance test this performance difference is significant. This demonstrates the positive impact of domain-specific MLM on hierarchical product classification. The performance increase of the model 2-rel_l-rnn compared to the models 2-rel_s-rnn and 2-rand_l indicates that a large number of related product offers improves the model's performance the most. Among the classification heads, the results of the models 2-rel_l-flat, 2-rel_l-hier and 2-rel_l-rnn show that the RNN profits most from domain-specific pre-training. Table **??** reveals that the models 1-rel_l-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.

### E.6 Effect of Using Web Shop Categories

The results in Table **??** indicate a slightly positive effect of using the heterogeneous categorization information from the original web shops during pre-training. A comparison of the models 2-rel_$s_{nocat}$-rnn and 2-rel_s-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_$s_{ext}$-rnn, the model's performance improves up to the performance level of the model 2-rel_l-rnn and 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 available text 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.

# F Related Work

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

## F.1 Domain Adaptation

The technique of pre-training transformer models on large text corpora and fine-tuning them for downstream tasks has proven successful in NLP [2, 3, 12]. 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 [2]. Through pre-training the model obtains the ability to encode natural language [13]. This knowledge about natural language is then transferred to downstream tasks. Pre-trained domain-specific models have shown that pre-training on domain-specific text improves the performance on downstream domain-specific tasks [4, 5, 7, 23]. 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 [5]. 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.

## F.2 Hierarchical Product Classification

Related work shows that exploiting the hierarchical structure can improve classification results [6, 8, 14–16]. The participants of the MWPD challenge show that in addition to exploiting the hierarchy, pre-trained transformer models can boost the results of hierarchical product classification tasks [11]. Team Rhinobird, the winners of the MWPD challenge, combine a pre-trained transformer model BERT with a hierarchical classification head [14]. Their Dynamic Masked Softmax classification head sequentially predicts the categories of different levels in the product hierarchy by actively restricting the classes, which can be predicted on the lower levels based on the predicted parent level node. Through different BERT based representations an ensemble of classifiers with the Dynamic Masked Softmax head enables Rhinobird to reach a wF1 score of 88.08 on the MWPD task that is used in this paper. Additionally, Rhinobird [14] applies pseudo labelling on the unlabeled test data to further improve the performance of their model. 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 [18]. Our best model outperforms ASVinSpace's approach by +2.14 wF1 points.

# G Conclusion

Our results show that the performance of transformer models on hierarchical product classification tasks can be improved through domain-specific pre-training on a corpus of related product offers. All domain-specifically pre-trained and fine-tuned models outperform the baseline model, which relies on general pre-training and task-specific fine-tuning. 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 we adjust the product offer representation during pre-training to exploit the special characteristics of the attributes title, description and category, the result on the hierarchical product classification task is further improved even though only a small corpus of related products is used for pre-training. 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.

# References


[1] A. Joulin, E. Grave, P. Bojanowski and T. Mikolov. Bag of Tricks for Efficient Text Classification. *15th Conference of the European Chapter of the Association for Computational Linguistics*, 2:427–431, 2017.

[2] J. Devlin, M. Chang, K. Lee and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 4171–4186, 2019.

[3] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and V. Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*, 2019.

[4] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. So and J. Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36:1234–1240, 2019.

[5] D. Zhang, Z. Yuan, Y. Liu, Z. Fu, F. Zhuang, P. Wang, H. Chen and H. Xiong. E-BERT: A Phrase and Product Knowledge Enhanced Language Model for E-commerce. *arXiv:2009.02835 [cs]*, 2020.

[6] D. Gao, W. Yang, H. Zhou, Y. Wei, Y. Hu and H. Wang. Deep Hierarchical Classification for Category Prediction in E-commerce System. *3rd Workshop on e-Commerce and NLP (ECNLP)*, 64–68, 2020.

[7] I. Beltagy, K. Lo and A. Cohan. SciBERT: A Pretrained Language Model for Scientific Text. *Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3615–3620, 2019.

[8] C. Silla and A. Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22:31–72, 2011.

[9] R. Meusel, A. Primpeli, C. Meilicke, H. Paulheim and C. Bizer. Exploiting Microdata Annotations to Consistently Categorize Product Offers at Web Scale. *International Conference on Electronic Commerce and Web Technologies*, 83–99, 2015.

[10] S. Kiritchenko, S. Matwin and A. Famili. Functional Annotation of Genes Using Hierarchical Text Categorization. *ACL Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, 2005.

[11] Z. Zhang, C. Bizer, R. Peeters and A. Primpeli. MWPD2020: Semantic Web challenge on Mining the Web of HTML-embedded product data. *CEUR Workshop Mining the Web of HTML-embedded Product Data*, 2720:2–18, 2020.

[12] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov and Q. Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *33rd Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[13] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li and P. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Machine Learning Research*, 21:1–67, 2020.

[14] L. Yang, E. Shijia, S. Xu and Y. Xiang. Bert with Dynamic Masked Softmax and Pseudo Labeling for Hierarchical Product Classification. *CEUR Workshop Mining the Web of HTML-embedded Product Data*, 2720:2020.

[15] J. Wehrmann, R. Cerri and R. Barros. Hierarchical Multi-Label Classification Networks. *35th International Conference on Machine Learning*, 5075–5084, 2018.

[16] R. You, Z. Zhang, Z. Wang, S. Dai, H. Mamitsuka and S. Zhu. AttentionXML: Label Tree-based Attention-Aware Deep Model for High-Performance Extreme Multi-Label Text Classification. *33rd Conference on Neural Information Processing Systems (NeurIPS)*, 32:11, 2019.

[17] Z. Zhang and M. Paramita. Product Classification Using Microdata Annotations. *International Sematic Web Conference (ISWC)*, 716–732, 2019.

[18] J. Borst, E. Krner and A. Niekler. Language Model CNN-driven similarity matching and classification for HTML-embedded Product Data. *CEUR Workshop Mining the Web of HTML-embedded Product Data*, 2720:11, 2020.

[19] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou and H. Hon. Unified Language Model Pre-training for Natural Language Understanding and Generation. *33rd Conference on Neural Information Processing Systems (NeurIPS)*, 2019.

[20] A. Primpeli, R. Peeters and C. Bizer. The WDC Training Dataset and Gold Standard for Large-Scale Product Matching. *International World Wide Web Conference (WWW)*, 381–386, 2019.

[21] T. Dietterich Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10:1895–1923, 1998.

[22] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma and R. Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *International Conference on Learning Representations (ICLR)*, 2020.

[23] S. Gururangan, A. MarasoviÄ‡, S. Swayamdipta, K. Lo, I. Beltagy, D. Downey and N. Smith. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. *58th Annual Meeting of the Association for Computational Linguistics*, 8342–8360, 2020.