2019 IEEE 13th International Conference on Semantic Computing (ICSC)

End-to-End Product Taxonomy Extension from Text Reviews

Longquan Jiang*, Or Biran†, Mrigank Tiwari*, Zhiliang Weng* and Yassine Benajiba‡
*Mainiway, †Elemental Cognition, ‡Symanto
*{longquan.jiang,mrigank,zhiliang.weng}@mainiway.com, †orb@elementalcognition.com,
‡yassine.benajiba@symanto.net

***Abstract*—Product ontologies - consisting of a taxonomic categorization of product types and lists of attributes that types and products have - are invaluable for analyzing sales, opinions and ratings of items on e-commerce sites. Unfortunately, many international and smaller sites lack such ontologies, and instead feature only coarse high-level categories. We present a Siamese neural model which utilizes such coarse categories to learn a fine-grained hierarchical categorization of products, and jointly extract lists of product attributes from text reviews. We show that our model retains a high accuracy on the categorization task for unseen products and unseen category depths, and as a side effect learns to extract useful product attributes.**

I. Introduction

Some modern e-commerce sites, such as Amazon, have an internal fine-grained classification of products and an ontology of product attributes. These are useful for a wide variety of applications, such as opinion mining and sentiment analysis, personalized product ratings and recommendations, sales predictions and targeted ads.

Many international sites and smaller retailers, however, have only a coarse high-level categorization of products and no structured information about product attributes, which acts as a barrier for many useful applications. In this paper, we propose an end-to-end model that jointly learns a fine-grained product taxonomy and extracts product attributes from free text product reviews and a coarse single-level categorization of products.

Our model employs a Siamese network architecture with attention over an LSTM applied to the input text. The model learns a categorical affinity score using the known coarse categories, and we show that this score then applies to lower-level categories which the model has not been exposed to. At the same time, the attention layer learns to focus on terms which are highly predictive of a product's category, and we show that these terms have a correspondence to attributes of the product, as well as other useful information such as category names.

We achieve very high accuracy on the categorization task using agglomerative clustering on the affinity matrix, and show that unseen, new products can then be matched to the right category (including unseen low-level categories) by using the same model to compare them with existing clusters. In a human evaluation, we also show relatively high accuracy on the attribute extraction task. Both the fine-grained categorization and attribute extraction tasks are learned without any labels, as a side effect of the model trained on coarse categorization.

[†] [‡] Work done in collaboration with Mainiway AI Lab

II. Related Work

While there is not much work on product taxonomies specifically, there is a large body of literature on taxonomy building, extraction and extension. WordNet [1], a manually-constructed taxonomic thesaurus, has been widely used in the Natural Language Processing community, but is relatively small and sparse. In response, taxonomies and ontologies have been extracted from Wikipedia, utilizing its unique structure [2]–[4] or from generic text, using lexical cues to extract semantic relations [5]–[7].

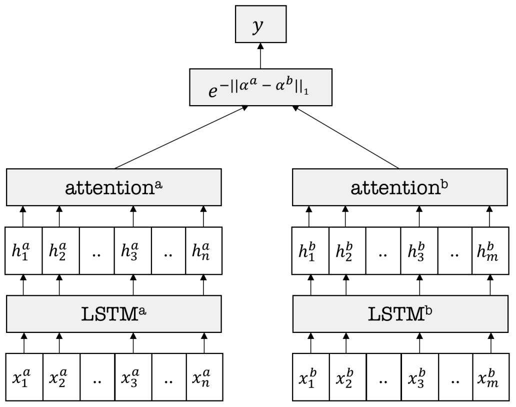
Another line of research, more conceptually similar to our work, focuses on fully supervised models for direct taxonomic relation classification [8], [9]. These approaches use generic (but hand-crafted) features, so the model learns to focus on the taxonomic relation between two concepts instead of the semantic specifics of the concepts. Our Siamese network similarly learns to identify a particular taxonomic relationship (siblings) between two products.

In addition, our work is related to the task of product attribute extraction, which often ties together the extraction of attribute names, values, and opinions [10]–[13].

III. Method

The intuition behind our approach is that product reviews written by humans contain references to specific attributes of a product - for example, the screen of a smartphone, the zipper of a suitcase, or the material quality of a pair of shoes. Crucially, these attributes are usually distributed along the lines of product categories - indeed, they have a crucial part in defining at least some categories. High-heeled shoes are a subcategory of shoes, where all member products share an attribute (high heel) which is not shared by other shoes. By comparing the reviews of two products and focusing on these discriminative words, we should be able to learn whether or not the two products are of the same category. Our hypothesis is that while learning to match products with the same top-level

978-1-5386-6783-5/19/$31.00 ©2019 IEEE
DOI 10.1109/ICSC.2019.00045



y
$e^{-||\alpha^a - \alpha^b||_1}$
attention^a
attention^b
h_1^a h_2^a .. h_3^a .. h_n^a
h_1^b h_2^b .. h_3^b .. h_m^b
LSTM^a
LSTM^b
x_1^a x_2^a .. x_3^a .. x_n^a
x_1^b x_2^b .. x_3^b .. x_m^b

Fig. 1. The model architecture

category, a model will also learn to extract these attributes and to find more subtle differences between products (fine-grained categories).

Siamese networks, designed as a way to compare two objects, were first introduced by [14]. By focusing on comparison, the network is not tied to a particular set of classes. A Siamese network can be diagrammed as having two branches (although both branches share the same parameters), both feeding to one layer that performs some distance measure to get an output. The idea behind this architecture is to learn a transformation function that transforms the input representation into a space where similar objects have similar representations.

When the input objects consist of text of variable length, a natural way to represent them is with a Recurrent Neural Network (RNN). The first to employ this variant, to our knowledge, were [15], who use LSTM Siamese networks to compare pairs of sentences. We use a similar architecture, with two major differences. First, instead of classification, we train our network to learn a soft affinity function; we then use this affinity to build a full hierarchical clustering of the products, which allows an arbitrarily fine-grained categorization. Second, we add an attention layer [16] on top of the LSTM in order to focus on discriminative terms. Our architecture is shown in Figure III.

The attention layer utilizes a context vector as described in [17]. Specifically, given an LSTM hidden state h_i^a in branch a of the network, we derive the final attended value of the branch, α^a, as follows:

$$u_i^a = tanh(W_w h_i^a + b_w) \tag{1}$$

$$\alpha_i^a = \frac{exp(u_i^{aT} u_w)}{\sum_i exp(u_i^{aT} u_w)} \tag{2}$$

$$\alpha^a = \sum_i \alpha_i^a h_i^a \tag{3}$$

and α^b for the second branch is derived in the same way. Finally, the affinity between the branches is measured as:

$$y = e^{-||\alpha^a - \alpha^b||_1} \tag{4}$$

We train the model using the contrastive loss function [18]. After training the model on all pairs of products in the training set (our data set is described in Section IV), we use the affinity matrix - the matrix of scores predicted by the model for each pair of products - to perform hierarchical agglomerative clustering on the entire training set. Using this clustering, for any arbitrary number of clusters k, we extract the k medoids of those clusters. The medoid of a cluster c is the product with the maximum total affinity score to other products in the cluster:

$$\mu_c = \underset{j \in c}{\arg\max} \sum_{i \in c} S(i,j) \tag{5}$$

where $S(i,j)$ is the affinity score assigned to products i and j by the model. For a previously unseen product p, we choose its cluster in a k-clustering by comparing it with the k medoids and choosing the max:

$$cluster(p) = \underset{c=1\ldots k}{\arg\max}\, S(p, \mu_c) \tag{6}$$

By changing the number of clusters k we can choose an arbitrarily fine-grained categorization of products.

As a side effect of finding an affinity score between two products, the model also give us a list of terms from the reviews of the products which are candidate attributes. We extract the attention scores for all words in the input and treat those with a score higher than a tunable threshold τ as candidate attributes.

A. Attribute Filtering

Naturally, not all candidates will correspond to attributes: the model learns to attend to all discriminative terms, which include product attributes as well as other terms related to a product's category. Other discriminative terms include common recipients of products (e.g. "wife" is common for perfumes, while "father" is common for watches); sentiment words ("beautiful" for watches or shoes but not smartphones or perfumes); adjectives describing attribute values ("fast" for smartphones, "soft" for suitcases); etc.

To alleviate this problem, we use the NTUSD sentiment lexicon [19] to remove sentiment terms, and the HowNet taxonomic lexicon [20] to remove terms that are not nouns, and terms in 11 unlikely categories[1]. We also remove numeric terms and terms of only one Chinese character.

IV. Data Set

To evaluate our approach, we collected a new data set of products and reviews from a leading Chinese e-commerce site. We will make the data available to researchers upon publication. The data set was created by crawling over 180,000 reviews of products from six (coarse-grained)

[1]"Place", "idea", "group", "knowledge", "artifact", "space", "business", "issue", "cause", "information", "food", and "feeling".

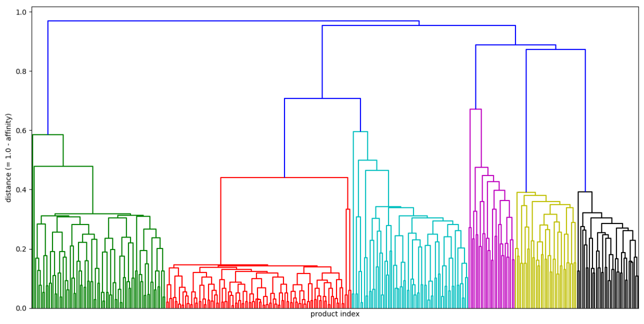


Fig. 2. The dendogram of 6-clustering the training set

		6-clust.	18-clust.
Train	Siamese	98.3%	N/A
Test	Embeddings	24.32%	30.93%
	Att. vectors	35.48%	49.17%
	Siamese	87%	87.8%

TABLE I
Accuracy (Rand index) achieved in the categorization task

	Precision			Atts /
τ	Anno 1	Anno 2	Avg.	item
0.0001	33.4%	47%	40.2%	7.66
0.001	62.7%	66.2%	64.4%	1.85
0.005	67.2%	67.6%	67.4%	1.02
0.01	68.6%	68.9%	68.8%	0.54

TABLE II
Precision achieved in the attribute extraction task

categories: perfumes, purses, suitcases, watches, smart phones and shoes.

The data set was split into a training set of 353 products with $25,000$ reviews each, and a test set of 125 products with $5,000$ reviews each. Both contain products from all six categories in approximately similar distribution.

We had four human annotators provide gold labels (as free text) for fine-grained categories within each top-level category. Where there was disagreement, we asked the annotators to make a decision amongst themselves on the final label. Agreement was fairly high: in over 55% of cases, at least three annotators chose the exact same label; in most other cases, the "disagreement" was only a different choice of synonym or a slightly more detailed description. In all cases, arriving to an agreement was quick and easy. The final number of fine-grained categories found in the test set is 18.

V. Experiments

To prepare the data for consumption by the model, we randomly sampled a small number n of reviews for each product. In our experiments we set $n = 20$; lower values of n result in significantly smaller attribute lists and larger values increase run time without significant gains. The sampled reviews were segmented by Jieba[2], converted to word-level precomputed fastText [21] embeddings, and then concatenated and fed to the LSTM as a single long sequence. The longest textual representation of a product was 1132 characters.

We trained the model on all possible pairs of products from the training set, for 20 epochs. At the end of the training period, we used the affinity matrix of all product pairs in an average-linkage agglomerative clustering. The dendogram for this clustering is shown in Figure V, with the top 6 clusters highlighted (corresponding approximately to the 6 high-level categories). The accuracy (Rand index) of the 6-clustering on the training set is 98.3%.

As described in Section III, we proceed to compute the k medoids for each k-clustering in order to match the test set products to existing clusters. We show results for two values of k: 6, the number of coarse-grained categories; and 18, the number of fine-grained categories found by the human annotators.

We compare our methods with two baselines. The first uses the averaged embeddings of the inputs directly to represent the products. In the second, a list of attributes for each product is first extracted from the reviews using POS patterns; the products are then represented as TF-IDF vectors of attribute mentions. For both baselines, we use the same agglomerative clustering algorithm - with these vectors instead of the affinity scores - to generate baseline k-clusterings. The results (Rand index) are shown in Table I.

In addition to evaluating our model on the categorization task, we evaluate the list of extracted attributes. For each product, we showed the list of terms with high attention scores to two human annotators and asked them to mark as correct terms which really are attributes of the product and as incorrect terms that are not. The Cohen's kappa inter-annotator agreement between the two annotators was $\kappa = 0.46$

The results are shown in Table II. Since we do not have complete ground truth attribute lists, and so cannot provide a true recall measure, we instead show the precision of the extracted list and the average number of attributes extracted per product.

As demonstrated by the results in Table I, the learned affinity score does more than discriminate between the original 6 categories. Instead, the network learns a semantic category distance that remarkably holds for lower-level, unseen categories, so that we are able to use a straightforward clustering technique to build a fine-grained taxonomy below the original high-level categories. Unseen products can be matched to existing clusters efficiently and with high accuracy by using the same model to compare them with a single sample from each cluster (the medoid). For deeper hierarchies, we expect that instead of comparing with every single cluster we can instead compare only

[2]https://github.com/fxsjy/jieba

among the clusters of each level, and iteratively do the same for lower levels; this approach would make the matching process more efficient when the number of categories is very large. We leave this experiment to future work.

The attribute extraction experiments summarized in Table II show that the threshold τ is an important hyperparameter that controls the trade off between higher precision and the number of extracted attributes. The sweet spot for attribute extraction seems to be with τ somewhere in the range of $0.001 - 0.005$; beyond that, gains in precision are smaller while the number of attributes decreases significantly. However, it is important to keep in mind that τ was not tuned on a validation set (we would have had to perform multiple human evaluations to do that), so the results in Table II represent the only experiment we have for comparing different values of τ.

To illustrate the need for the filtering step, some examples of terms with very high attention scores are sentiment terms like 很好 - "very good" and 足够 - "adequate"; intensifiers such as 非常 - "extremely" and 很 - "very"; and *values* of attributes rather than attribute labels, e.g. 好看 - "good looking" and 简单大方 - "simple and elegant", referring to the style of the suitcase. However, once those categories are filtered out, many of the remaining terms with high scores are on the mark: 质量 ("quality"), 图案 ("pattern"), 款式 ("style"), 容量 ("capacity") and 大小 ("size"), all of which are good examples of discriminating attributes. There are also less discriminating attributes which the model rejects: 价格 ("price") receives a very low attention score, likely because it repeats in every category.

In addition to extracting attributes, the high attention terms can be useful in other ways. For example, one prominent term is 包包 - "bag", which appears in many suitcase / purse reviews, but not in shoes (or other categories). This term is directly related to the label of the product's category. In fact, it turns out that a product's fine-grained taxonomic label (produced by human annotators unexposed to these lists, as described in Section IV) is extracted by the attention layer as an exact match for 48.8% of products in the test set and as a soft match (allowing one character added or removed) for 58.4% of products. We do not further explore label extraction in this paper, and leave it to future work. Attention scores also tend to be high for sentiment words (which we filter out explicitly). While these are not attributes, extracting sentiment or opinions may be another useful side effect of the model in a different configuration.

VI. Conclusion

We described a Siamese neural model which utilizes an attentive LSTM encoder and learns a categorical semantic affinity function that can be used to construct fine-grained taxonomies of products from coarse categories. As a side effect, the model highlights useful terms which can be used for attribute extraction, as well as category labels and opinions (although we do not explore these last two options). While this work is in its early stages, we demonstrate the potential of this approach and view it as a step towards end-to-end extraction of product ontologies. In future work, we intend to develop our approach towards this goal, as well as look into extracting more general taxonomies and ontologies from different types of text, beyond products and reviews.

References

[1] C. Fellbaum, *WordNet.* Wiley Online Library, 1998.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web.* Springer, 2007, pp. 722–735.

[3] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web.* ACM, 2007, pp. 697–706.

[4] S. P. Ponzetto and R. Navigli, "Large-scale taxonomy mapping for restructuring and integrating wikipedia," in *IJCAI*, 2009.

[5] R. Girju, A. Badulescu, and D. Moldovan, "Learning semantic constraints for the automatic discovery of part-whole relations," in *NAACL-HLT*, 2003.

[6] R. Snow, D. Jurafsky, and A. Y. Ng, "Semantic taxonomy induction from heterogenous evidence," in *ACL*, 2006.

[7] M. Baroni and A. Lenci, "Distributional memory: A general framework for corpus-based semantics," *Computational Linguistics*, vol. 36, no. 4, pp. 673–721, 2010.

[8] Q. X. Do and D. Roth, "Constraints based taxonomic relation classification," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 2010.

[9] O. Biran and K. McKeown, "Classifying taxonomic relations between pairs of wikipedia articles," in *IJCNLP*, 2013.

[10] B. Liu, M. Hu, and J. Cheng, "Opinion observer: analyzing and comparing opinions on the web," in *Proceedings of the 14th international conference on World Wide Web.* ACM, 2005.

[11] R. Ghani, K. Probst, Y. Liu, M. Krema, and A. Fano, "Text mining for product attribute extraction," *ACM SIGKDD Explorations Newsletter*, vol. 8, no. 1, pp. 41–48, 2006.

[12] K. Shinzato and S. Sekine, "Unsupervised extraction of attributes and their values from product description," in *IJCNLP*, 2013.

[13] P. Petrovski and C. Bizer, "Extracting attribute-value pairs from product specifications on the web," in *Proceedings of the International Conference on Web Intelligence.* ACM, 2017.

[14] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," in *NIPS*, 1994.

[15] J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

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

[17] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[18] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Computer vision and pattern recognition, 2006 IEEE computer society conference on*, vol. 2. IEEE, 2006, pp. 1735–1742.

[19] L.-W. Ku and H.-H. Chen, "Mining opinions from the web: Beyond relevance retrieval," *Journal of the American Society for Information Science and Technology*, vol. 58, no. 12, pp. 1838–1850, 2007.

[20] Z. Dong, Q. Dong, and C. Hao, "Hownet and its computation of meaning," in *Proceedings of the 23rd International Conference on Computational Linguistics: Demonstrations*, 2010.

[21] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *CoRR*, vol. abs/1607.04606, 2016.