Generative Models for Product Attribute Extraction

Ansel Blume[*,1], **Nasser Zalmout**[2], **Heng Ji**[2], **Xian Li**[2]
[1]University of Illinois Urbana-Champaign, [2]Amazon
blume5@illinois.edu, {nzalmout,jihj,xianlee}@amazon.com

Abstract

Product attribute extraction is an emerging field in information extraction and e-commerce, with applications including knowledge base construction, product recommendation, and enhancing customer experiences. In this work, we explore the use of generative models for product attribute extraction. We analyze their utility with hard and soft prompting methods, and demonstrate their ability to generate implicit attribute values, which state-of-the-art sequence tagging models are unable to extract. We perform a wide range of experiments on Amazon and MAVE product attribute datasets, and are the first to present results on multilingual attribute extraction. Our results show that generative models can outperform state-of-the-art tagging models for explicit product attribute extraction while having greater data efficiency, that they have the unique ability to perform implicit attribute extraction, and that in certain settings large language models can perform competitively with finetuned models with as little as two in-context examples.

1 Introduction

E-commerce has exploded in recent years, with large online retailers offering billions of products and shipping millions of packages per day. With such a large number of offerings, having a complete set of each product's properties is imperative for effective retrieval (search), product analytics, and recommendations (Zalmout et al., 2021). However, the sheer number of offerings makes building product profiles with this metadata a challenge—filling in large sets of product properties is arduous for sellers, and new fields for existing products must be filled retroactively. Clearly, automated methods are necessary for scalable extraction.

Product attribute extraction is designed to address this challenge. In this task, a product profile with text and possibly visual data is provided to a model, along with a pre-defined attribute whose value is to be determined. These attributes represent key product characteristics, such as the size of a shirt or the scent of a candle. The goal is for the model to determine the set of attribute values from the product profile, or to indicate that no value can be found if such information cannot be inferred (Xu et al., 2019; Yan et al., 2021; Wang et al., 2020a; Yang et al., 2021). Extracted product attributes can be used for a variety of purposes, including analyzing data to better understand the product offerings, recommending relevant products for customers, and providing easy access to distinguishing product information.

Figure 1: An example product listing with extracted product attributes highlighted. The attribute extraction models take as input the product title and product description (not pictured), then attempt to produce values for a specified attribute based on the product text.

Existing efforts frame product attribute extraction as sequence labeling (Zheng et al., 2018; Xu et al., 2019; Yan et al., 2021; Yang et al., 2021) or extractive question answering tasks (Wang et al., 2020a; Ding et al., 2022). While these approaches yield high precision (the answers are necessarily grounded in the text), they cannot discover attribute values that are implied but not explicitly mentioned in the text. For example, a table described as having a "wavy grain" can be inferred to be made out of wood. However, unless the word *wood* appears

[*] Work done during an internship at Amazon.

Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 575–585
December 6-10, 2023

in the text, sequence labeling and extractive question answering architectures are unable to extract this value for the *material* attribute.

Generative models do not suffer from this limitation. Such models are not constrained to pointing to tokens in the input text, and instead produce free-form text by generating tokens autoregressively. This enables generative models to conceivably generate any value, encountered in the input or not. At the same time, this freedom can result in hallucinations, where language models produce inaccurate or imagined values (Maynez et al., 2020; Zhou et al., 2021; Li et al., 2021; Ji et al., 2022).

Motivated by this flexibility, we aim to address three research questions: 1. How do generative models compare to state of the art sequence-tagging models on product attribute extraction? 2. To what extent are generative models able to produce implicit attribute values? 3. How does the input prompt affect extraction performance?

In this work, we apply generative language models to product attribute extraction. We define the task of implicit attribute extraction, and show that generative models can extract implicit attributes while outperforming or achieving comparable performance to state-of-the-art sequence labeling architectures on the MAVE (Yang et al., 2021) dataset and monolingual and multilingual Amazon data. We show that generative models are especially effective in low-resource scenarios, and demonstrate that in certain settings large language models (LLMs) can perform as well as finetuned models with as few as two in-context examples.

2 Related Work

The task of product attribute extraction has become increasingly relevant with the rise of e-commerce (Zheng et al., 2018; Yang et al., 2021; Xu et al., 2019; Yan et al., 2021). Performing closed-vocabulary attribute extraction, where product attribute values are selected from a fixed set, can be realized as a classification problem (Ghani et al., 2006). However, such methods are less scalable as the number of products increases, and newly introduced products can contain never-before-seen attribute values. To address this, (Zheng et al., 2018) pioneered the task of open-vocabulary attribute extraction, where attribute values are not restricted to a fixed set. (Zheng et al., 2018) achieves this by using a sequence tagging architecture to mark tokens that are values for a given attribute.

Since (Zheng et al., 2018), state-of-the-art models have followed extractive paradigms such as sequence tagging or extractive question answering. (Xu et al., 2019) scales up the OpenTag model of (Zheng et al., 2018) by training a single model to handle all attributes, instead of training one model per attribute. (Wang et al., 2020a) improves upon the architecture of (Xu et al., 2019) and frames the attribute extraction problem as extractive question answering. (Yan et al., 2021) returns to sequence tagging, but uses hypernetworks and a mixture of experts to personalize model parameters for each attribute without needing to train a separate network for each. Finally, (Yang et al., 2021) introduces the MAVE dataset and the MAVEQA model, which uses the ETC (Ainslie et al., 2020) long-document model to handle large product profiles in a sequence-tagging paradigm.

Three prior approaches have applied generative models to product attribute extraction. (Roy et al., 2021) formulates the extraction problem as text-infilling and generation tasks, generating attribute values from the product title. (Roy et al., 2022) generates attributes present in the text and their corresponding values, instead of using the attribute as as query. (Lin et al., 2021) performs multimodal attribute extraction with image and textual inputs. Our work differs from those prior in that we distinguish between implicit and explicit attribute extraction and consider generative models' ability to produce implicit values, we evaluate generative models on multilingual data, we provide results for large language models, and we explore the importance of different prompt setups.

3 Method

3.1 Problem Formalization

The product attribute extraction problem may be formalized as follows: given product text t and an attribute to extract a, the goal is to produce a set of strings $\{s_1, \ldots, s_n\}$ which indicate the product's attribute values for a based on the text t. These values may be surface mentions—that is, substrings of t. For example, given product text for a decorated Christmas tree including the profile's title and description, and the attribute *color* to extract, any mentions of colors in the product text, e.g. "red" and "gold", should be extracted. Such mentions are called *explicit* attribute values, as they occur directly in the text. However, attribute values may not be explicitly mentioned: for example, the tree

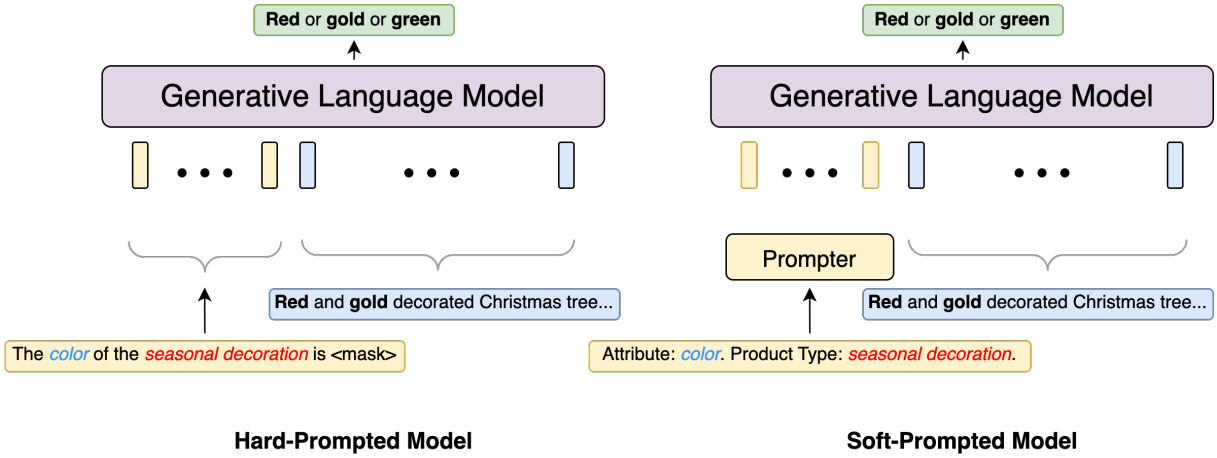

Figure 2: The hard and soft prompting architectures. The models receive the product text along with a prompt detailing the attribute to extract and the product type (category). Blue denotes the input text, and yellow indicates the hard prompt or dynamically generated soft-prompt embeddings. Note that generative models can produce the value **green** for the *color* attribute, despite the word **green** not occurring in the product text.

may be decorated in red and gold, but the tree itself is green. A model should ideally identify *green* as a value for the attribute *color*, regardless of whether the word *green* appears in the product text. We call such values *implicit* attribute values as they are implied but not explicitly mentioned by the text, and this task *implicit attribute value extraction*.

3.2 Generative Framework

Our framework for product attribute extraction uses a generative language model which takes as input the product text, product type (category), and the attribute to extract. The model predicts the attribute values from the input by autoregressively generating text. We specify the product type and attribute to extract by in-filling a template, then passing this text in with the product text.

Product text may have multiple distinct values to extract. To represent these values, we join them with the word "or". So a product with attribute values A and B would have the generation target "A or B". The generated text is post-processed to extract the attribute values. During training, the values are ordered as they occur in the product text for consistency and training stability.

3.3 Prompting

One advantage of generative models is their ability to receive natural language (hard) or soft (embedding) prompts that can better elicit language model's parametric knowledge than unstructured inputs. Tuning such natural language or soft prompts with a frozen language model has been shown to generalize well to few-shot tasks and rival full fine-tuning (Liu et al., 2021; Li and Liang, 2021; Lester et al., 2021; Tu et al., 2022). Due to prompts' potential importance, we explore two different prompt setups for our medium-sized models—a term we use to refer to any model not considered a "large language model" (LLM).

As product attribute extraction is a task significantly different from language models' pretraining objectives, prompting frozen medium-sized language models does not produce viable results. Therefore, we finetune the medium-sized models on the product data and leave the LLMs frozen, using prompts for both to indicate the product type and the attribute to be extracted. For the medium-sized models, we experiment with hard and soft–prompted architectures.

3.3.1 Hard Prompt

For hard prompts, we provide the model with manually defined natural language templates which are in-filled with the product type and attribute information (Figure 2, left). Hard prompts for the medium-sized language models have a <mask> token which represents the attribute value(s) to be extracted. The intent is for the prompt to resemble the language model's pretraining objective—having the model denoise the masked text—in order to perform our desired task.

As the large language models are frozen, we provide them with a task description and two in-context examples (Brown et al., 2020) of the attribute to be extracted (Section A.4).

3.3.2 Soft Prompts

Typical soft prompting approaches prepend a sequence of learned embeddings to the input text which are tuned in place of the entire model (Liu et al., 2021; Gao et al., 2021; Li and Liang, 2021; Lester et al., 2021). As mentioned previously, we tune the medium-sized language model parameters, but we also maintain a separate prompter module (Figure 2, right) which generates prefix embeddings that are conditioned on the attribute and product type (Liu et al., 2021; Levine et al., 2022). We choose BERT (Devlin et al., 2019) as our prompter network and freeze all but its top two layers.

4 Experiments

In this section we present our experiments on product attribute extraction with generative models. We start by comparing generative models to sequence-tagging architectures on implicit and explicit attribute extraction. Next, we demonstrate that generative frameworks continue to perform well on multilingual attribute extraction. Finally, we scale up the number of attributes with explicit attribute extraction on the MAVE dataset.

All models are evaluated using precision and recall via set-wise comparison. Precision is the likelihood that a predicted attribute value extraction is in the set of ground truth values, and recall is the likelihood that a ground-truth attribute value is predicted by the model. Models may tag no values or output *unknown* to indicate no values are present. Additional details can be found in Section A.1.

For all experiments, we use the 7B parameter versions of the Llama (Touvron et al., 2023) and conversation-tuned Vicuna (Chiang et al., 2023) large language models. We provide these models with a task description and two in-context examples, followed by the product profile to extract from. This saturates the models' context windows of 2048 tokens. Training details for the finetuned models can be found in Section A.3.

4.1 Monolingual Attribute Extraction

To determine the efficacy of generative models for implicit attribute extraction, we evaluate sequence-tagging and generative models on an in-the-wild Amazon product dataset. Unlike the MAVE dataset which is constructed primarily by an ensemble of sequence-tagging models, the attribute values in this English Amazon dataset are filled by sellers and do not need to occur in the product text. We choose SUOpenTag and AdaTag (Xu et al., 2019; Yan et al., 2021) as our tagging models. For generative models, we evaluate the finetuned hard- and soft-prompted BART architectures, along with the hard-prompted frozen LLMs. Dataset statistics can be found in Appendix Table 5.

4.1.1 Explicit Attribute Extraction

Table 1 shows the results. As sequence tagging models can only produce values found in the text, we separately evaluate all models on explicit values. Even in this setting, generative models are competitive with the tagging baselines. Surprisingly, the soft-prompted model achieves significantly higher precision than the tagging models, despite tagging models receiving negative examples from non-tagged tokens during training. Individual examples show that the soft-prompted model was likely to ground its answers in the text, resulting in its higher precision on explicit examples than its hard-prompted counterpart. AdaTag, the better performing tagging model, made most of its errors tagging words which were reasonable attribute values, but which were incorrect in the context of the attribute or product. For example, AdaTag tagged *coffee* as the *scent* for a clove conditioner, when coffee was used for coloring instead of for its scent (see Figure 3 for additional such examples). This suggests that generative models have a better contextual understanding of the attributes, as they are less likely to tag values which are reasonable only in isolation.

4.1.2 Implicit Attribute Extraction

Considering performance on both implicit and explicit examples, we see that the tagging models' recall drops precipitously as they are unable to predict implicit values. The hard-prompted model does not ground its extractions as often as the soft-prompted model, and so gets a larger number of implicit values correct. However, the soft-prompted model is also capable of generating implicit values—manual inspection of a random sample to account for synonyms shows that the soft-prompted model achieves .29 precision for these implicit values. See Figure 3 for qualitative examples on implicit attribute extraction.

4.2 Multilingual Attribute Extraction

To further analyze the effectiveness of generative models, we train and evaluate multilingual generative models on an in-house multilingual dataset

	Explicit			Implicit + Explicit		
	Precision	Recall	F1	Precision	Recall	F1
AdaTag	.5145	**.4226**	.4640	.3978	.3497	.3722
SUOpenTag (BERT)	.4312	.3827	.4055	.4005	.3219	.3569
BART (Hard Prompt)	.4643	.3903	.4241	**.4343**	.3570	.3919
BART (Soft Prompt)	**.7131**	.4195	**.5282**	.3982	**.3862**	**.3921**
Llama* (Hard Prompt)	.5484	.3205	.4046	.2996	.3447	.3206
Vicuna* (Hard Prompt)	.4973	.2326	.3170	.3520	.2065	.2603

Table 1: Results on the English Amazon dataset, evaluated on examples with only explicit attribute mentions and on those with both implicit and explicit mentions. * indicates the model was frozen.

	en_US		fr_FR		de_DE	
	Precision	Recall	Precision	Recall	Precision	Recall
SUOpenTag (mBERT)	.4244	.3184	.5158	.2050	.3678	.2145
SUOpenTag (XLM-base)	.4323	.3104	.5193	.1936	.3761	.2126
SUOpenTag (XLM-large)	**.4705**†	.3050	**.5633**†	.1942	**.3982**	.2081
mT5-small (Hard Prompt)	.4167	.3145	.4866	.1781	.3810	.2650
mBART-50 (Hard Prompt)	.3701	**.3279**†	.4084	**.2062**†	.3533	**.2839**†
Llama* (Hard Prompt)	.2315	.2580	.2386	.1142	.1604	.1623
Vicuna* (Hard Prompt)	.2936	.1239	.1094	.0371	.1514	.0652
mT5-small (Translated Prompt)	.3900	.2450	.5073	.1569	.4219†	.2393
mBART-50 (Translated Prompt)	.3445	.2926	.3883	.1923	.3408	.2657
Llama* (Translated Prompt)	.2435	.2535	.2186	.0872	.2068	.1575
Vicuna* (Translated Prompt)	.2931	.1242	.0923	.0455	.1424	.0525

Table 2: Results on the Multilingual Amazon dataset. The top set of models receive English prompts, regardless of the product text's language. The bottom set translates the prompt into the product text's language. **Bold** indicates the best scores in the upper set of rows, and † indicates the best scores across all rows. * indicates the model was frozen.

containing English, French, and German examples, with the number of training examples decreasing in that order (Appendix Table 6). We train mBART-50 and mT5-small (Tang et al., 2020; Xue et al., 2020) with hard prompts as our generative models, and use SUOpenTag (Xu et al., 2019) with mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2019) as our sequence-tagging baselines (AdaTag is based on Glove (Pennington et al., 2014) and is English-only). The Llama and Vicuna LLMs are prompted as before.

We hypothesize that as the models were pretrained with single-language instances, using single-language inputs better matches their pretraining objectives. To determine whether code-switching affects performance, we use prompts translated into the product text's language (prompts are in English by default), training variants of the mBART-50 and mT5-small models on these translations and providing the LLMs with these prompts.

The results are shown in Table 2. Surprisingly, the single language–input models (Translated Prompt) perform slightly worse than those with the code-switched inputs, and the LLMs perform comparably. We speculate that the medium-sized models perform worse as they saw more English data during pretraining, and therefore perform better with English inputs. The LLMs' performance is limited most by the number of in-context examples, so the prompt language makes little difference.

The generative models achieve significantly higher recall on German, the lowest resource language, than the tagging models. These results are consistent with generative models' better performance in Section 4.1—they appear to have a better understanding of the attributes and so require fewer training samples.

	Input Text	Target Attribute	Label	Prediction
BART (Soft Prompt)	GHOST Glow: Beauty and **Detox Support Formula - 30 Servings**, Passionfruit - Skin Boosting Biotin...	Item Form	powder	powder
	ABN Oxygen Sensor Socket with Side Wire Cutout...**constructed from** drop forged heat-treated **chrome-vanadium steel**...	Material	metal	metal
	Bath and Body Works Deep Cleansing Hand Soap, Kitchen Lemon, **8 fl. oz**. Lot of 2...	Item Form	liquid	bar
AdaTag	Lavanila Body Butter. All-Natural Moisturizing Body Butter With **Vanilla Scent**...	Scent	vanilla	vanilla
	AiXiAng Cute Mini 24 Pieces Little Elephant...delightfully **detailed soap**...Packaged in a small, beautifully detailed **gift box**...	Item Form	bar	box
	Ray-Ban RX5154 Clubmaster Square Prescription **Eyeglass Frames**...Each pair of Ray-Ban optical frames come with a cleaning **cloth** and case...	Material	plastic	cloth

Figure 3: Examples from the English Amazon dataset with correct and incorrect predictions. **1.** BART learns that nutritional supplements are likely to be powders, and this example fits that prototype, despite "powder" being implicit. **2.** The label "metal" is unmentioned, but can be inferred from the bolded text. **3.** BART is unable to infer that the product is a liquid from the "8 fl. oz" measurement. **4.** AdaTag is able to extract the explicitly mentioned scent. **5, 6.** The attribute values are implicit, but AdaTag still incorrectly outputs attribute values for secondary objects (see Section 4.1.1).

	Low Resource		Medium Resource		High Resource		All	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
AdaTag	.3910	**.8815**	.6491	.8045	.8341	**.9410**	.5919	**.8627**
SUOpenTag (BERT)	.3369	.5188	.3264	.6023	.4652	.7579	.3602	.6159
BART (Hard Prompt)	**.8454**	.7964	**.9040**	.8190	**.9576**	.9245	**.8994**	.8385
BART (Soft Prompt)	.8136	.7751	.8927	**.8324**	.9498	.9342	.8816	.8405
Llama* (Hard Prompt)	.2658	.4936	.2024	.3950	.1962	.4745	.2245	.4448
Vicuna* (Hard Prompt)	.2207	.4097	.2418	.3942	.2664	.3357	.2392	.3845

Table 3: Results on the MAVE dataset after splitting it into low, medium, and high resource attributes and performing stratified-sampling. **Bold** indicates the best performance in a column, and * that the model was frozen.

4.3 Explicit Attribute Extraction on MAVE

Finally, we evaluate explicit attribute extraction on a subset of the MAVE dataset (Yang et al., 2021), a product attribute dataset created by an ensemble of sequence tagging AVEQA models (Wang et al., 2020b). The MAVE dataset's unique advantage is its large number of attributes. To consider how the amount of training data affects performance, we categorize the attributes in MAVE by how many examples they have: low (< 500), medium ($500 \leq \ldots \leq 5000$), and high-resource (> 5000). We then sample 1/10th of the attributes from each of these strata, obtaining 28 low, 27 medium, and 14 high-resource attributes. We consider the same models as in the monolingual Amazon data[1].

Table 3 shows the results. The hard and soft-prompted generative models perform comparably, and experience little performance degradation across resource levels. On the other hand, consistent with the results in the multilingual setting, the sequence tagging models struggle more on the low-resource attributes than their generative alternatives, indicating that generative models have better low-resource generalization. Plots of model performance as a function of each attribute's number of training examples (Figure 4) suggest that the fine-tuned generative models outperform the tagging

[1]We trained the MAVEQA model using the code released for (Yang et al., 2021) but were not able to obtain good results.

models across resource levels.

4.4 Performance of Large Language Models

While the LLMs' performance is limited by the number of examples that fit in their context windows, they perform surprisingly well on explicit attribute extraction on the English Amazon dataset: Vicuna and Llama achieve higher precision than half and three fourths of the finetuned models respectively, while obtaining respectable recall. However, on MAVE the finetuned medium-sized models far outperform the prompted LLMs, emphasizing the usefulness of finetuning. The difference in results on the MAVE and Amazon datasets is due to the distribution shift between the Amazon dataset's train and test splits: the attribute values in the training data are filled by sellers, and do not conform to strict rules; the Amazon test data is manually annotated by a team of annotators following rigid instructions to ensure high quality. On the other hand, MAVE's data is completely generated by an ensemble of AVEQA (Wang et al., 2020b) models, so the train and test splits follow the same distribution. This indicates that frozen LLMs have the opportunity to outperform finetuned medium-sized models, especially under distribution shift, as their task definition and pretraining makes them less reliant on training examples.

Comparing the LLMs to one another, we find that despite Vicuna's conversation tuning and impressive performance compared to much larger models (Zheng et al., 2023), Llama performed almost universally better on all three datasets. This performance difference was not only by way of extraction quality, but was also due to Llama's adherence to the prompt's specified generation format (see Section A.4). Vicuna answered in various conversational formats, not following instructions, hence making answer extraction from the generated text more difficult.

5 Conclusion

In this work, we demonstrated the benefits of generative models for product attribute extraction. We showed that generative models can outperform state-of-the-art sequence tagging models on explicit attribute extraction and, unlike sequence tagging models, can perform implicit attribute extraction. Next, we demonstrated that generative models perform better in low-resource settings than tagging models on both multilingual data and the MAVE dataset. Finally, we showed that large language models can perform well with as few as two in-context examples, emphasizing generative models' data efficiency. For future work, we plan to scale up our soft prompt architecture to large language models, and push the limits of their performance with long context lengths to provide additional in-context examples.

References

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. ETC: Encoding Long and Structured Inputs in Transformers.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. *arXiv:2005.14165 [cs]*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised Cross-lingual Representation Learning at Scale. In *Annual Meeting of the Association for Computational Linguistics*.

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]*.

Yifan Ding, Yan Liang, Nasser Zalmout, Xian Li, Christan Grant, and Tim Weninger. 2022. Ask-and-Verify: Span candidate generation and verification for attribute value extraction. In *EMNLP 2022*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making Pre-trained Language Models Better Few-shot Learners. *ArXiv*, abs/2012.15723.

Rayid Ghani, Katharina Probst, Yan Liu, Marko Krema, and Andrew Fano. 2006. Text mining for product attribute extraction. *SIGKDD Explor. Newsl.*, 8(1):41–48.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2022. Survey of hallucination in natural language generation. *ACM Computing Surveys*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The Power of Scale for Parameter-Efficient Prompt Tuning.

Yoav Levine, Itay Dalmedigos, Ori Ram, Yoel Zeldes, Daniel Jannai, Dor Muhlgay, Yoni Osin, Opher Lieber, Barak Lenz, Shai Shalev-Shwartz, et al. 2022. Standing on the shoulders of giant frozen language models. *arXiv preprint arXiv:2204.10019*.

Chenliang Li, Bin Bi, Ming Yan, Wei Wang, and Songfang Huang. 2021. Addressing Semantic Drift in Generative Question Answering with Auxiliary Extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 942–947, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, abs/2101.00190.

Rongmei Lin, Xiang He, Jie Feng, Nasser Zalmout, Yan Liang, Li Xiong, and Xin Luna Dong. 2021. PAM: Understanding Product Images in Cross Product Category Attribute Extraction. In *Proceedings of the 27th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, volume 2657, pages 1–9. CEUR-WS.

Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. 2021. GPT Understands, Too.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Kalyani Roy, Pawan Goyal, and Manish Pandey. 2021. Attribute Value Generation from Product Title using Language Models. In *Proceedings of The 4th Workshop on E-Commerce and NLP*, pages 13–17, Online. Association for Computational Linguistics.

Kalyani Roy, Tapas Nayak, and Pawan Goyal. 2022. Exploring Generative Models for Joint Attribute Value Extraction from Product Titles.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual Translation with Extensible Multilingual Pretraining and Finetuning.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothee Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models.

Lifu Tu, Caiming Xiong, and Yingbo Zhou. 2022. Prompt-Tuning Can Be Much Better Than Fine-Tuning on Cross-lingual Understanding With Multilingual Language Models. In *Conference on Empirical Methods in Natural Language Processing*.

Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D. Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. 2020a. Learning to Extract Attribute Value from Product via Question Answering: A Multi-task Approach. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, pages 47–55, New York, NY, USA. Association for Computing Machinery.

Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. 2020b. Learning to extract attribute value from product via question answering: A multi-task approach. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 47–55.

Huimin Xu, Wenting Wang, Xin Mao, Xinyu Jiang, and Man Lan. 2019. Scaling up Open Tagging from Tens to Thousands: Comprehension Empowered Attribute Value Extraction from Product Title. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5214–5223, Florence, Italy. Association for Computational Linguistics.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2020. mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer. In *North American Chapter of the Association for Computational Linguistics*.

Jun Yan, Nasser Zalmout, Yan Liang, Christan Earl Grant, Xiang Ren, and Xin Dong. 2021. AdaTag: Multi-Attribute Value Extraction from Product Profiles with Adaptive Decoding. In *Annual Meeting of the Association for Computational Linguistics*.

Li Yang, Qifan Wang, Zac Yu, Anand Kulkarni, Sumit K. Sanghai, Bin Shu, Jonathan L. Elsas, and Bhargav Kanagal. 2021. MAVE: A Product Dataset for Multi-source Attribute Value Extraction. *Proceedings of*

the Fifteenth ACM International Conference on Web Search and Data Mining.

Nasser Zalmout and Xian Li. 2022. Prototype-representations for training data filtering in weakly-supervised information extraction. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 467–474, Abu Dhabi, UAE. Association for Computational Linguistics.

Nasser Zalmout, Chenwei Zhang, Xian Li, Yan Liang, and Xin Luna Dong. 2021. All you need to know to build a product knowledge graph. KDD '21, page 4090–4091, New York, NY, USA. Association for Computing Machinery.

Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. OpenTag: Open attribute value extraction from product profiles. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1049–1058. Association for Computing Machinery.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Chunting Zhou, Graham Neubig, Jiatao Gu, Mona Diab, Francisco Guzmán, Luke Zettlemoyer, and Marjan Ghazvininejad. 2021. Detecting Hallucinated Content in Conditional Neural Sequence Generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1393–1404, Online. Association for Computational Linguistics.

A Appendix

A.1 Evaluation Details

For sequence tagging models, we define a prediction as a contiguous sequence of "I"-tagged tokens in an "I/O" tagging scheme, whereas a null prediction occurs if no tokens are tagged. For generative models, we define a prediction as any "or"-separated words besides a single *unknown* token, and a null prediction by having the model output an *unknown* token. All precision and recall values are macro averages over the evaluated attributes. Postprocessing is applied to extract the attribute values from the generated text in the case of multiple generated values.

Text which exceeds the model input length is truncated for the medium-sized models. For the LLMs, text which exceeds the input length is treated as an "unknown" prediction.

A.2 Dataset Details

A.2.1 Amazon Datasets

Table 5 and Table 6 show the dataset statistics for the Amazon datasets. A separate validation set was split off from 20% of the training data for model selection. The training datasets are derived from seller's attribute annotations, whereas the test datasets are curated by a team of annotators. The train datasets contain a single attribute value per example, whereas the test datasets can contain multiple values per example, representing multiple acceptable answers. Refer to (Zalmout and Li, 2022) for more details on the processing setup for this dataset.

Each locale has five attributes in their training and evaluation datasets. The English Amazon dataset contains the Material, Scent, Item Form, and Flavor, and Fabric Type attributes. For the multilingual dataset, all three locales share the Material, Scent, Item Form, and Flavor attributes, all in each locale's respective languages. Each locale also has an additional, unique attribute: en_US with FabricType, fr_FR with Recommended Uses for Product, and de_DE with Color.

We use SUOpenTag initialized with different multilingual encoders on the Multilingual Amazon dataset instead of AdaTag, as the AdaTag model is based on English Glove embeddings (Pennington et al., 2014) and is not inherently multilingual.

A.2.2 Stratified Mave Subset

As described in Section 4, we divide the MAVE dataset into low, medium, and high–resource attributes. We then sample one tenth of the attributes from each of the divisions to ensure that our subset is representative of the overall MAVE dataset—directly sampling from the examples would favor high-resource attributes as these examples dominate the dataset, forcing the medium resource attributes towards the low end, and low-resource attributes towards zero examples. We define our strata by attributes having fewer than 500 examples, between 500 and 5000 examples, and greater than 5000 examples.

A.3 Training details

On the English Amazon dataset, we use learning rates of 2e-5 for all models besides AdaTag, which benefits significantly from a higher learning rate of 3e-4. On the Multilingual Amazon dataset, we use a learning rate of 2e-5 for all models. On MAVE ,

[Prompt]: The <attribute> of the <product_type> is <mask> [Title]: ... [Description]: ...
[Prompt]: The <attribute> of the <product_type> in <locale> is <mask> [Title]: ... [Description]: ...

Table 4: The input templates used for the hard-prompted, medium-sized generative models. The templates are used for monolingual and multilingual data, respectively. The *product_type* is a known category from a taxonomy to which each product belongs; we include this to improve extraction performance. The <*attribute*>, <*product_type*>, and <*locale*> tokens are in-filled before being passed to the model.

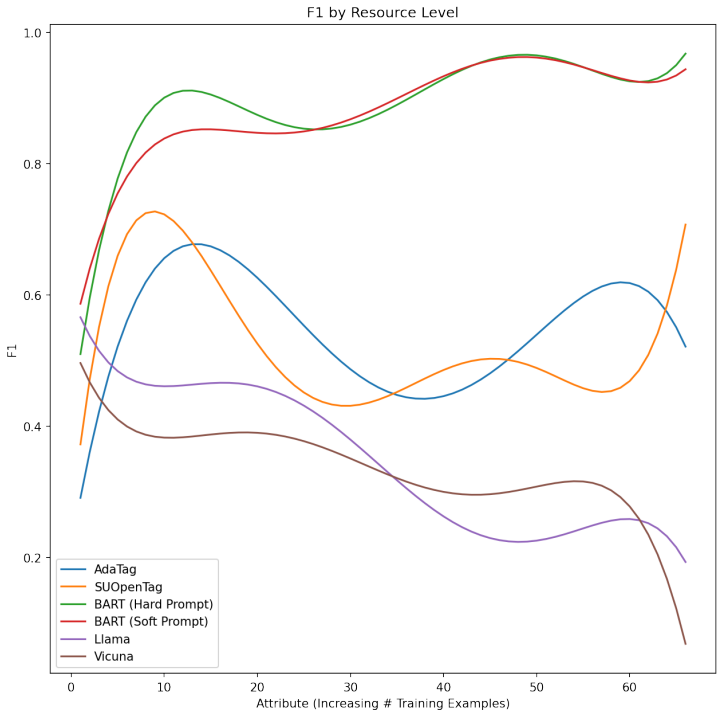

Figure 4: The models' F1 scores on each attribute of the MAVE subset, with the attributes sorted in increasing order by number of training examples. Curves are approximated with polynomial regression with interpolation for attributes with no predictions. Some variation is due to differing attribute difficulty, despite the increasing number of training examples (e.g. the frozen Llama and Vicuna models' downward trends).

	Explicit	Implicit	Unknown
Train	72.5 (152798)	.50 (1050)	27 (57073)
Test	52.9 (2885)	7.8 (427)	39.3 (2139)

Table 5: Dataset statistics for the English Amazon dataset used for implicit and explicit attribute extraction. The first numbers indicate approximate percentages across a row, and the parenthesized numbers are the raw counts.

	en_US	fr_FR	de_DE
Train	38 (210895)	32 (174507)	30 (164972)
Test	30 (5451)	40 (7107)	30 (5564)

Table 6: Dataset statistics for the multilingual Amazon dataset. The first numbers indicate approximate percentages, and the parenthesized numbers are the raw counts.

we use a learning rate of 3e-5 for all models besides AdaTag, which again uses a learning rate of 3e-4. We use the Adam (Kingma and Ba, 2014) optimizer for all models.

We train models on the English and Multilingual Amazon datasets for 100K steps and on the Stratified MAVE Subset for three epochs, evaluating on a holdout set and selecting the best models based on their validation losses. We use batch sizes of 16 for the Amazon datasets and 8 for MAVE.

The input prompt for the soft-prompted BART's prompter module is padded to 20 tokens in order to pass a fixed number of prompt embeddings to BART.

A.4 Large Language Model Prompt

To prompt the Llama and Vicuna large language models, we provided a task description followed by two in-context examples of the extraction task for the given attribute. For the experiments on translating the input text, we translated the entirety of the prompt besides the product text and answers. We experimented first with a non-conversational prompt for the Llama model, which described the task but lacked the "interactiveness" of saying that the model would be provided with examples, followed by the input to be extracted from. Instead, the prompt provided the example products with answers delimited by `[Answer]` and `[/Answer]` tokens, followed by the product to be extracted from, with no distinction between the provided examples and the product input. Surprisingly, this performed significantly worse on the non-conversational Llama than spelling out each example, question, and answer conversationally in our final prompt shown in Figure 5. We therefore used this prompt for both Llama and Vicuna.

Determine values for the specified attributes based on the product text. If there are multiple values, separate them by "or", as in "value 1 or value 2". If there are none, write "unknown". Say only the specified value(s), or unknown, and nothing else. I will provide one or more examples, followed by the input for which you say the answer.

Example 1: [Title]: ... [Description]: ... [Bullets]: ...
Question: The **item form** of the **detergent** is?
Answer: **actionpacs or pacs**

Example 2: [Title]: ... [Description] ... [Bullets]: ...
Question: The **item form** of the **skin cleaning agent** is?
Answer: **gel**

Input: [Title] ... [Description]: ... [Bullets]: ...
Question: The **item form** of the **gift wrap** is?
Answer:

Figure 5: The chosen large language model prompt. Bolded phrases vary depending on the in-context example. Ellipses indicate the in-filled product text.