ASTRA: Automatic Schema Matching using Machine Translation

Tarang Chugh and Deepak Zambre
Amazon
{tchugh, dzzambre}@amazon.com

Abstract

Many eCommerce systems source product information from millions of sellers and manufactures, each having their own proprietary schemas, and employ schema matching solutions to structure it to enable informative shopping experiences. Meanwhile, state-of-the-art machine translation techniques have demonstrated great success in building context-aware representations that generalize well to new languages with minimal training data. In this work, we propose modeling the schema matching problem as a neural machine translation task: given product context and an attribute-value pair from a source schema, the model predicts the corresponding attribute, if available, in the target schema. We utilize open-source seq2seq models, such as mT5 and mBART, fine-tuned on product attribute mappings to build a scalable schema matching framework. We demonstrate that our proposed approach achieves a significant performance boost (15% precision and 7% recall uplift) compared to the baseline system and can support new attributes with precision $\geq 95\%$ using only five labeled samples per attribute.

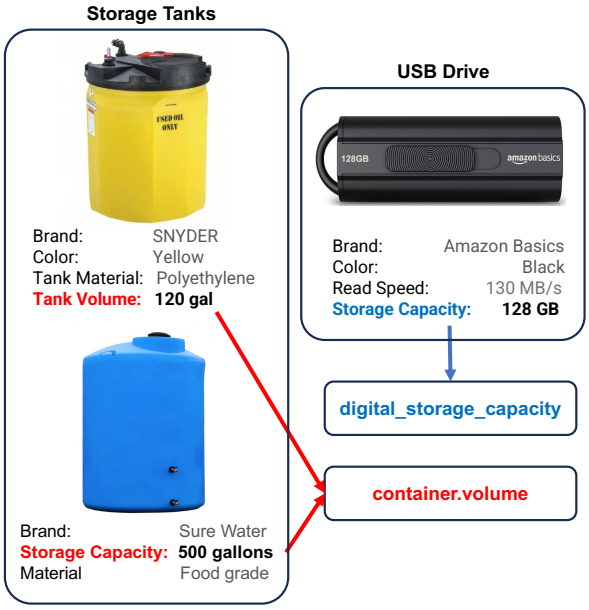

Figure 1: Different vendors may represent semantically similar product facts using different attribute names (*e.g. Tank Volume and Storage Capacity*). Conversely, same attribute name (*e.g. Storage Capacity*) could be used to represent two distinct logical attributes for different product types (*e.g. Storage Tanks and USB Drive*).

1 Introduction

eCommerce retailers rely heavily on structured catalogs containing essential product information to provide best-in-class customer experiences such as faceted product search, personalized recommendations, and valuable product insights. However, consolidating product data into a structured catalog involves integrating information from various heterogeneous data sources such as manufacturer fact sheets, brand websites, and GDSN feeds[1] (Zheng et al., 2018). These sources often present data in diverse schema representations across product categories, languages, and feed types as illustrated in Figure 1.

Due to the sheer scale of product offerings, it is prohibitively expensive and time-consuming to manually curate a comprehensive product catalog for eCommerce systems like Amazon, Walmart, etc. Typically, each system operates with its own unique proprietary schema, necessitating that sellers (*e.g.* manufacturers or distributors) adhere to specific schema constraints and complicates listing management for the sellers. To address this, eCommerce systems typically employ automatic schema matching models to consolidate product information from disparate sources and simplify listing experiences. Figure 2 illustrates a high-level view of the schema transformation pipeline supporting a

[1]GDSN stands for Global Data Synchronization Network. It is a network of data pools that allows businesses to share high-quality product information seamlessly with their trading partners. https://www.gs1.org/services/gdsn

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track, pages 1237–1244

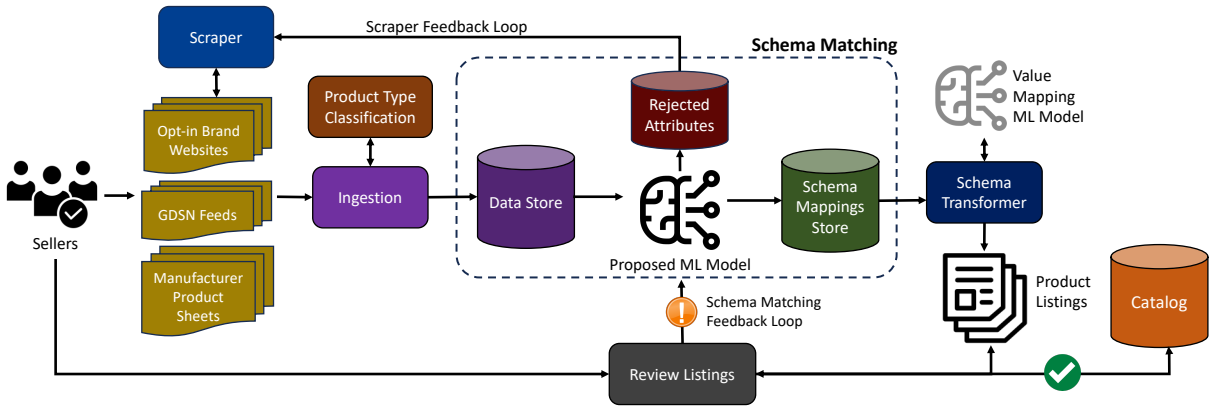

Figure 2: An overview of the automatic product listing creation pipeline utilizing a neural machine translation model for automatic schema matching. Sellers provide their product data in heterogenous formats which is automatically schema mapped and validated to be contributed to the catalog.

listing experience. Simplifying and automating the listing process encourages sellers to onboard their entire selection of products, thereby improving the overall shopping experience for customers.

Existing schema matching approaches may exhibit several critical limitations, such as: (i) address schema matching as a closed-set problem, which renders them unable to handle source attributes that cannot be mapped to an existing target attribute, (ii) train one model per attribute which require a large amount of labeled training data per attribute, (iii) require significant efforts, such as architectural changes or model re-training, to support new attributes, (iv) limit model input to the attribute key-value pairs, which may potentially lack critical product context, and (v) inefficient pairwise comparisons of embedding representations. These limitations underscore the challenges faced by current schema matching methodologies and highlight areas where improvements are necessary to enhance model flexibility, efficiency, and contextual understanding.

Parallels can be drawn between the task of schema matching and neural machine translation, which has recently achieved state-of-the-art performance in several NLP tasks, such as language translation, text summarization, and question answering (Stahlberg, 2020). Just as machine translation converts text between languages while preserving semantics, schema matching identifies correspondences of product attributes from one schema to another while preserving the intended information. For instance, mapping the attribute *Tank Volume* from one manufacturer's schema to *item_volume* in a target schema is analogous to translating the English word *hello* to *hola* in Spanish. Both processes require understanding context and meaning of the original term to ensure accurate and useful translation in the target format to address the problem of *impedance mismatch* (Ireland et al., 2009).

In this work, we propose to leverage the machine learning techniques used in language translation to effectively and efficiently align diverse product data sources to a standardized target schema, facilitating faster and accurate product listings. Inspired by similarities between machine translation and schema mapping, we propose ASTRA (Automatic Schema Matching via Machine Translation), a generative approach to perform schema matching for product entities in the eCommerce domain that scales for thousands of product types as well as disparate sources of data. Our main contributions are summarized as:

- proposed a novel framework to model schema matching as a generative neural machine translation task,
- addressed critical limitations of existing frameworks, such as open-set schema matching and extending attribute coverage without requiring any changes to the model architecture,
- proposed e-commerce specific components and optimizations, like vocabulary augmentation, token budgeting, and confidence score proxy, to achieve high precision schema matching, and
- demonstrated scalability of the approach for extending to new attributes with few shot learning.

2 Related Work

Traditionally, schema mapping approaches assumed structured databases from a handful number of sources with clean data (Miller et al., 2000; Rahm and Bernstein, 2001). However, such techniques are not suited to large-scale eCommerce data, where the number of available domain schemas is in the order of millions when accounting for different manufacturers, vendors, and product categories.

In recent literature, approaches have relied on attribute value extraction to enrich product listings. These approaches train supervised models to extract missing attribute values from free text, such as product title and product description, using multi-class classification (Ghani et al., 2006), neural sequence labeling (Zheng et al., 2018; Xu et al., 2019; Yan et al., 2021) or extractive question answering (Wang et al., 2020; Ding et al., 2022). However, such approaches are better suited for products with unstructured text (i.e. title, bullet points, descriptions) and not directly applicable to schema matching in situations where product information is available in semi-structured form such as websites or product feeds. Additionally, since these approaches are often trained at attribute or category level, achieving scale is difficult in settings where there exist a large number of constantly evolving applicable attributes and categories.

Recent studies have investigated using state-of-the-art pre-trained large language models (LLMs) for attribute value extraction in a question-answering framework (Blume et al., 2023; Brinkmann et al., 2023; Baumann et al., 2024). However, it is not only prohibitively expensive to extract each attribute value using LLMs at a product level, they are also prone to hallucinations (Jiang et al., 2024), producing outputs that are not grounded to the input data.

On the other hand, unsupervised techniques for schema matching have leveraged Word2Vec (Nozaki et al., 2019; Kolyvakis et al., 2018) and FastText (Shieh et al., 2021) to generate learned representations of source and target attribute key-values and computed semantic similarity to perform schema mapping. While unsupervised approaches scale well with large number of attributes and categories, they are unable to achieve the required precision for hands-off-the-wheel schema matching.

3 Method

3.1 Problem Formalization

The problem of attribute matching may be formalized as follows: given an input attribute a_s from a source schema S, the goal is to identify an attribute a_t, if it exists, from a target schema T. Each attribute a may be characterized by a key (*i.e.* attribute name) k and a set of values V. For our use-case, we assume that our models will be trained to match an unspecified number of source schemas to a single fixed target schema (which in our case is the Amazon product schema).

3.2 Schema Mapping Framework

Attribute schema mapping and machine translation share significant similarities in their fundamental processes. Both involve transforming input data from one structured format to another while preserving the inherent meaning and intent. Our schema mapping framework uses neural machine translation models to learn and infer product attribute correspondences between various external source schemas and a known target schema. Specifically, we employ transformer-based multilingual sequence-to-sequence (seq2seq) generation models, namely mT5 and mBART, leveraging self-attention mechanisms to generate context-aware schema mappings.

We model schema matching as a translation task, where the input token sequence contains serialized product information, including *product type*, *attribute name* and *attribute value* from the source schema. The output token sequence is the corresponding attribute, if available, from the target schema. To map a source attribute to the correct target attribute, it is crucial to use the context (product information) to disambiguate between potential target attributes as illustrated in Figure 1.

3.2.1 mT5: Multi-lingual Text-to-Text Transfer Transformer

An mT5 model (Xue et al., 2020) is a multi-lingual variant (supports 101 languages) of T5 model (Raffel et al., 2020), a basic encoder-decoder Transformer architecture (Vaswani et al., 2017), pre-trained as a masked language model, where consecutive spans of input tokens are replaced with a mask token and the model is trained to reconstruct the masked-out tokens. This innovative design of T5 model allows it to be pre-trained on a massive corpus and then fine-tuned for specific tasks using

Serialized Input Text Sequence	Output Sequence
<*PT*> SUITCASE <*KEY*> Made in: <*VAL*> China	country_of_origin
<*PT*> COMPUTER_DRIVE_OR_STORAGE <*KEY*> Disk Speed (RPM) <*VAL*> 7200rpm	hard_disk rotational_speed
<*PT*> CAMERA_DIGITAL <*KEY*> Image Sensor Size <*VAL*> 35mm Full Frame (36 x 24 mm)	photo_sensor size
<*PT*> JUMP_STARTER <*KEY*> バッテリータイプ <*VAL*> リチウムイオンバッテリー	battery cell_composition
<*PT*> SHIRT <*KEY*> fabric <*VAL*> 85% Cotton / 15% Polyster	material_composition
<*PT*> PERSONAL_FRAGRANCE <*KEY*> This deodorant works so well <*VAL*> so much better	<NOMAP>

Table 1: Serialized input data including product context using special tokens <*PT*>, <*KEY*>, and <*VAL*>. Output sequence is either the expected target attribute or <*NOMAP*> if the model is expected to reject the attribute.

the same architecture, providing a unified solution for a wide range of applications such as translation, summarization, question answering, and text classification. Compared to previous SOTA sequence modeling approaches, T5 model leverages a transformer-based architecture to enable efficient parallel processing and advanced attention mechanisms, ensuring high performance and scalability.

3.2.2 mBART: Multi-lingual Bidirectional and Auto-regressive Transformer

An mBART model (Liu et al., 2020) is a multilingual variant of the BART model (Lewis et al., 2019), an encoder-decoder Transformer architecture (Vaswani et al., 2017), pre-trained as a denoising auto-encoder. In this setup, the input text is corrupted by masking out tokens or shuffling the order of tokens, and the model is trained to reconstruct the original text. It works well for comprehension tasks but is particularly effective when fine-tuned for text generation.

3.3 Data Pre-processing and Setup

3.3.1 Data Cleaning

Product data from heterogeneous sources (e.g. web scraping, GDSN feeds) often contains noise, such as whitespace characters, formatting symbols, and HTML tags. Additionally, attributes in the target schema can be represented in a nested format, like *battery.cell_composition* and *hard_disk.rotational_speed*, which differ from typical natural language text used in model pre-training. To address this, we use regular expressions to clean the data and replace dot notation (".") with a whitespace character[2], to produce text that closely resemble natural language. This preprocessing step enhances model efficiency by allocating more input bandwidth to the product data and enabling faster training.

3.3.2 Data Serialization

Seq2Seq models take a sequence of tokens as input and generate a sequence of tokens as output. For schema matching, the input sequence includes serialized product information: product type, source attribute, and source value. The output sequence is the target attribute. We use special tokens <*PT*>, <*KEY*>, and <*VAL*> as markers to assist the model in understanding the beginning of product type, attribute key, and attribute value, respectively, in the input text. Examples of serialized input data are shown in Table 1.

3.3.3 Vocabulary Augmentation

The product data and the target schema may contain complex eCommerce-specific attributes like *eu_spare_part_availability_duration* and *oem_equivalent_part_number* which needs to be tokenized before input to the model. We augment the tokenizer's existing vocabulary with the complex target attributes which do not need to be split into smaller tokens[3]. This allows the model to train faster by reducing tokenization complexity, improving context understanding, optimizing memory usage and ensuring consistent representation. Additionally, vocabulary augmentation allows us to extract a confidence score proxy (as explained in Section 3.3.6) and filter out low confidence token sequences (potential hallucinations), thereby enhancing precision.

3.3.4 Model Input

Token budgeting involves managing the distribution of tokens across input sequences to ensure that the model's capacity is effectively utilized without exceeding its maximum limit. Both mT5

[2]During inference, the whitespace characters in the model prediction are replaced back with the dot symbol (".") to generate the nested attributes.

[3]We add a total of 2402 new tokens, increasing the vocab size from 250112 to 252514 for *mT5* model, and from 250054 to 252456 for *mBART* model.

and mBART models have a maximum input token limit of 1024 tokens. However, for our datasets, the average token length per input sequence is approximately 80 tokens, with a maximum of 330 tokens due to some longer attribute values. To ensure efficient model training, we limit the maximum token length to 128 tokens. This covers 98% of the training data, while we truncate the attribute values in the remaining sequences that exceed this length. The serialized input sequence containing product type, source attribute and source value(as shown in Table 1) is passed to the *generate* method of *MBartForConditionalGeneration* and *MT5ForConditionalGeneration* for *mBART* and *mT5* models, respectively.

3.3.5 Model Fine-tuning

Each fine-tuning experiment was run for a maximum of 20 epochs with evaluation during training enabled, using a validation set, for every N steps and early stopping patience of 10, where $N = 8000/batch\_size$. The model checkpoint with the lowest validation loss is saved and used for evaluation of the test set. We use a linear schedule with warm up for the learning rate adjustment for both mT5 and mBART. We utilize a $batch\_size$ of 16, 8, 2, and 4 for the *mT5-small*, *mT5-base*, *mT5-large*, and *mBART-50-large* models, respectively. All experiments[4] are conducted using the open-source SimpleTransformers[5] library.

3.3.6 Confidence Score Proxy

Our use case of automatic schema matching at scale requires a minimum precision of $\geq 95\%$. Therefore, it is essential to identify the confident model predictions and filter out the rest. Both mT5 and mBART models, similar to other transformer-based models such as BERT (Devlin et al., 2018) and GPT (Brown et al., 2020), do not inherently provide a confidence score with their predictions. These models generate output sequences token by token, selecting the most probable token at each position, but this probability is not usually exposed as a confidence score for the entire sequence. In our datasets, due to the vocabulary augmentation, the output sequences (*i.e.* target attributes) have a maximum length of two tokens, with over 80% of the target attributes represented by a single token.

[4]Experiments were conducted on a GPU linux server machine with 4 × 16GB Nvidia Tesla V100 GPUs running with CUDA version 12.2.

[5]Simple Transformers Library https://github.com/ThilinaRajapakse/simpletransformers

	Approved Attributes			Reject Attributes		
Approach	**P**	**R**	**F1**	**P**	**R**	**F1**
Baseline (mUSE)	0.83	0.44	0.58	-	-	-
mT5-small	0.94	0.48	0.64	0.92	0.40	0.56
mT5-base	0.98	0.51	**0.67**	0.96	0.42	0.58
mT5-large	0.98	0.49	0.65	0.96	0.43	**0.59**
mBART-Large-50	0.95	0.50	0.66	0.95	0.42	0.58

Table 2: Precision (P), Recall (R), and F1-score (F1) metrics of the proposed approach for automatic schema matching compared to baseline model on the D_{En} dataset. The baseline model did not have the capability to automatically reject the attributes.

This allows us to extract and utilize the logit scores of the predictions as a proxy for the model's confidence score. Post training, we utilize the validation set to determine the best score threshold to ensure precision $\geq 95\%$.

3.3.7 Handling Unavailable Attributes

Any source attribute that can be mapped to an existing attribute in the target schema is called an *Approved* attribute, while others that cannot be mapped to any available target attribute are considered as *Reject* attributes. This determination is made by subject matter experts, including product ontologists and trained auditors. *Reject* attributes include two cases: *Case-I*: input keys that does not represent any valid product information (*e.g.* incorrect scrape, non-product keys such as "Review rating"), and *Case-II*: input product attributes that can not be currently mapped to unavailable in the target schema. As shown in Table 1, the model is trained to handle *Case-I* attributes by utilizing a special token <*NOMAP*> as the target sequence. On the other hand, any model prediction (i) outside the set of valid target attributes, or (ii) within the set of valid target attributes but with a confidence score below a certain score threshold, learned from the validation dataset, is considered as a *Case-II* type of *Reject* attributes.

4 Experiments

4.1 Datasets

In this study, we utilize three datasets: D_{En}, D_{Multi}, and D_{HQ}. D_{En} contains 36,281 English-only samples (attribute key-value pairs) from more than 3,500 heterogenous schemas across 1,631 product types. These samples map to 2,824 unique product attributes in the target schema, averag-

	French			German			Italian			Japanese			Spanish		
Approach	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
Baseline (mUSE)	0.82	0.29	0.43	0.79	0.28	0.41	0.81	0.23	0.36	0.85	0.29	0.43	0.82	0.40	0.54
mT5-small	0.96	0.33	0.49	0.94	0.37	0.53	0.92	0.26	0.41	0.95	0.35	**0.51**	0.90	0.44	0.59
mT5-base	0.98	0.38	**0.55**	0.97	0.40	**0.57**	0.96	0.27	0.42	0.97	0.35	**0.51**	0.93	0.46	**0.62**
mT5-large	0.98	0.37	0.54	0.96	0.41	**0.57**	0.97	0.28	**0.43**	0.95	0.34	0.50	0.93	0.47	**0.62**
mBART-Large-50	0.96	0.38	0.54	0.96	0.40	0.56	0.95	0.26	0.41	0.97	0.35	**0.51**	0.92	0.43	0.59

Table 3: Precision (P), Recall (R), and F1-score (F1) metrics for the cross-lingual transfer learning capability when models are trained on English language and evaluated on five non-English languages, namely, French, German, Italian, Japanese and Spanish.

ing 12 labeled training samples per target attribute. D_{Multi} contains 993 samples in five non-English languages (French, German, Italian, Japanese, and Spanish) sourced from 38 schemas across 13 product categories. D_{HQ} contains $7,523$ high-quality samples, manually curated by ontologists, for two product categories (*DIGITAL CAMERA* and *SOFA*) containing 175 unique attributes. We use D_{En} and D_{Multi} to evaluate the performance of our proposed approach (ASTRA) in English and multilingual schema matching use-cases. We use D_{HQ} in our ASTRA-Lightning experiment to assess the approach's efficacy in supporting unseen attributes with only a few labeled samples.

4.2 Performance Evaluation Metrics

The model performance is evaluated for the two categories of *Approved* and *Reject* attributes[6] for the English dataset (D_{En}). The other two datasets D_{Multi} and D_{HQ} have been sourced from human-annotated tasks for model development, and contain only *approved* attributes. In our experiments, we use Precision, Recall, and F1-score metrics for performance evaluation.

4.3 Results

4.3.1 ASTRA: Automatic Schema Matching using Machine Translation

In this experiment, we evaluate the performance of neural machine translation models, mT5 and mBART, for schema matching. We fine-tune three variants of mT5: *mT5-small* ($300M$ parameters), *mT5-base* ($580M$ parameters, *mT5-large* ($1.2B$ parameters), and one variant of mBART: *mBART-Large-50* ($610M$ parameters). We use the D_{En} dataset containing English-only samples, with a 70/10/20 split for *train* / *validation* / *test* sets, ensuring no overlap of source schemas across the splits. For the baseline comparison, we fine-tune a multi-lingual Universal Sentence Encoder (mUSE) model (Yang et al., 2019), a dual-encoder architecture, and compute pairwise similarity between the learned embedding representations to match attributes. As presented in Table 2, the proposed *mT5-base* model achieves a 15% precision and 7% recall uplift compared to the baseline model for the *Approved* attributes. The baseline model, based on pairwise embedding similarity, could not exclude any *Reject* attributes, leading to significant manual labeling effort. The proposed approach can exclude such attributes with precision $\geq 95\%$ and recall $\geq 40\%$. We also evaluate the cross-lingual transfer learning capabilities of the models by testing the fine-tuned English models on the D_{Multi} dataset, containing five non-English languages, as unseen test set. As shown in Table 3, the overall best performing model, *mT5-base*, achieves an average F1-score increase of 10% over the baseline model.

4.3.2 ASTRA Lightning

In this experiment, we evaluate the hypothesis:- *Can the ASTRA model learn to map unseen attributes when fine-tuned using only a few training samples per target attribute?*, hence the term *lightning*. To test this, we use the D_{HQ} dataset (see Section 4.1 for details) containing over $7,500$ high-quality labeled samples from two product types and 175 unique target attributes.

To simulate unseen attributes, we removed occurrences of these 175 unique target attributes from the training data used to train the ASTRA model (D_{EN} dataset). This resulted in $25,344$ training samples (compared to $32,653$ samples in the original train-

[6] An *approved* attribute incorrectly excluded by the model causes funnel loss (*i.e.* preventing valuable product facts from being displayed) while incorrectly mapping a *reject* attribute to a target attribute results in poor customer experience.

Training Samples per Target Attribute (n)	Train Samples 25344+	Validation Samples (m)	Test Samples	Precision	Recall	F1	Area Under Curve (AUC)
Baseline	0	0	6019	-	-	-	0.5027
$n = 1$	95	95	6019	1	0.005	0.010	0.754
$n = 5$	475	190	6019	0.95	0.872	0.909	0.919
$n = 10$	950	190	6019	0.95	0.881	**0.914**	0.937

Table 4: Performance metrics to evaluate the minimum number of labeled samples required to onboard unseen product attributes to ASTRA model for auto-mapping.

ing data). We included $n \in \{1, 5, 10\}$ samples per target attribute in the training data for each experiment. The number of validation samples (m) used for early stopping is defined as $min(2, n)$ providing no more than two samples per target attribute. All remaining samples were used as test data.

We report four metrics: precision, recall, F1 score, and Area Under the Curve (AUC). We report the best model performance in maximizing recall, with the condition that precision $\geq 95\%$ (required for auto-mapping). If the model cannot achieve 95% precision, the AUC metric is included for comparison. Table 4 presents the performance metrics for onboarding unseen attributes. We observe that with just five labeled samples, the model achieves precision $\geq 95\%$ with high recall, meeting the requirements for auto-mapping.

5 Conclusions

This paper introduced application of neural machine translation to perform schema matching and showcased how this approach outperforms attribute embedding similarity based schema matching solutions. The performance evaluation experiment demonstrates effectiveness of vocabulary augmentation using product metadata, token budgeting and confidence score proxy for achieving reliable, consistent and precise schema matching. Finally, ASTRA Lightning, lays out a blueprint to extend the schema matching solution to new attributes with minimal new training data, thus making this approach suitable in cases where schema matching target is ever evolving.

References

Nick Baumann, Alexander Brinkmann, and Christian Bizer. 2024. Using llms for the extraction and normalization of product attribute values. *arXiv preprint arXiv:2403.02130*.

Ansel Blume, Nasser Zalmout, Heng Ji, and Xian Li. 2023. Generative models for product attribute extraction. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 575–585.

Alexander Brinkmann, Roee Shraga, and Christian Bizer. 2023. Product attribute value extraction using large language models. *arXiv preprint arXiv:2310.12537*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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 *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 110–110.

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

Christopher Ireland, David Bowers, Michael Newton, and Kevin Waugh. 2009. A classification of object-relational impedance mismatch. In *2009 First International Confernce on Advances in Databases, Knowledge, and Data Applications*, pages 36–43. IEEE.

Ling Jiang, Keer Jiang, Xiaoyu Chu, Saaransh Gulati, and Pulkit Garg. 2024. Hallucination detection in llm-enriched product listings. In *Proceedings of the Seventh Workshop on e-Commerce and NLP@ LREC-COLING 2024*, pages 29–39.

Prodromos Kolyvakis, Alexandros Kalousis, and Dimitris Kiritsis. 2018. Deepalignment: Unsupervised ontology matching with refined word vectors. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1-6 June 2018.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Renée J Miller, Laura M Haas, and Mauricio A Hernández. 2000. Schema mapping as query discovery. In *International conference on very large data bases*.

Kenji Nozaki, Teruhisa Hochin, and Hiroki Nomiya. 2019. Semantic schema matching for string attribute with word vectors. In *2019 6th International Conference on Computational Science/Intelligence and Applied Informatics (CSII)*, pages 25–30. IEEE.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Erhard Rahm and Philip A Bernstein. 2001. A survey of approaches to automatic schema matching. *the VLDB Journal*, 10:334–350.

Evan Shieh, Saul Simhon, Geetha Aluri, Giorgos Papachristoudis, Doa Yakut, and Dhanya Raghu. 2021. Attribute similarity and relevance-based product schema matching for targeted catalog enrichment. In *2021 IEEE International Conference on Big Knowledge (ICBK)*, pages 261–270. IEEE.

Felix Stahlberg. 2020. Neural machine translation: A review. *Journal of Artificial Intelligence Research*, 69:343–418.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. 2020. 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, Xinnian 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.

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. *arXiv preprint arXiv:2010.11934*.

Jun Yan, Nasser Zalmout, Yan Liang, Christan Grant, Xiang Ren, and Xin Luna Dong. 2021. Adatag: Multi-attribute value extraction from product profiles with adaptive decoding. *arXiv preprint arXiv:2106.02318*.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, et al. 2019. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*.

Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. Opentag: Open attribute value extraction from product profiles. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1049–1058.