Automating Opinion Extraction from Semi-Structured Webpages: Leveraging Language Models and Instruction Finetuning on Synthetic Data

Dawid Adam Plaskowski[1][a], Szymon Skwarek[1][b], Dominika Grajewska[1][c], Maciej Niemir[1][d] and Agnieszka Ławrynowicz[2][e]

[1]*Łukasiewicz - Poznań Institute of Technology, Poznań, Poland*

[2]*Poznań University of Technology, Poznań, Poland*

Keywords: Language Models, Information Extraction, Opinion Mining.

Abstract: To address the challenge of extracting opinions from semi-structured webpages such as blog posts and product rankings, encoder-decoder transformer models are employed. We enhance the models' performance by generating synthetic data using large language models like GPT3.5 and GPT-4, diversified through prompts featuring various text styles, personas and product characteristics. Different fine-tuning strategies are experimented, training both with and without domain-adapted instructions, as well as, training on synthetic customer reviews, targeting tasks such as extracting product names, pros, cons, and opinion sentences. Our evaluation shows a significant improvement in the models' performance in both product characteristic and opinion extraction tasks, validating the effectiveness of using synthetic data for fine-tuning and signals the potential of pretrained language models to automate web scraping techniques from diverse web sources.

1 INTRODUCTION

Acquiring customer review data by web scraping is a labor intensive task specific to use cases, as it requires developers to understand the structure of the document object model (DOM) of the targeted website and decide on the appropriate tools available. Although there are many resources available to scrape customer reviews from prominent e-commerce sites (Myers and McGuffee, 2015; Sharma, 2014), these resources are susceptible to structural alterations. Regular web changes, such as A/B testing, often modify the underlying DOM structure and making human written web scrapers ineffective. The task becomes even more challenging when the objective is to collect data from a diverse range of websites. This motivates research on different web scraping approaches that can be more robust and adaptable to website structural diversity and regular updates.

In this paper, we propose a transformer (Vaswani et al., 2017) web scraper specifically tailored for opinion mining from webpages that describe and compare multiple e-Commerce products within a single blog post.

Our approach is presented in Fig. 1. To extract valuable information, we require the concrete product name to appear in the context, as it is essential for assessing the exact product under analysis. To overcome the challenges of restrictive context window lengths in language models and the noisy web structure, we employ targeted HTML chunking. This approach enables us to link product names and their corresponding reviews more effectively. Our solution leverages a text classification-based algorithm, allowing us to segment valuable text chunks, linking them to specific product names. Further opinion extraction is performed by a pre-trained language model that is instructed to extract information about the pros and cons mentioned in the segment.

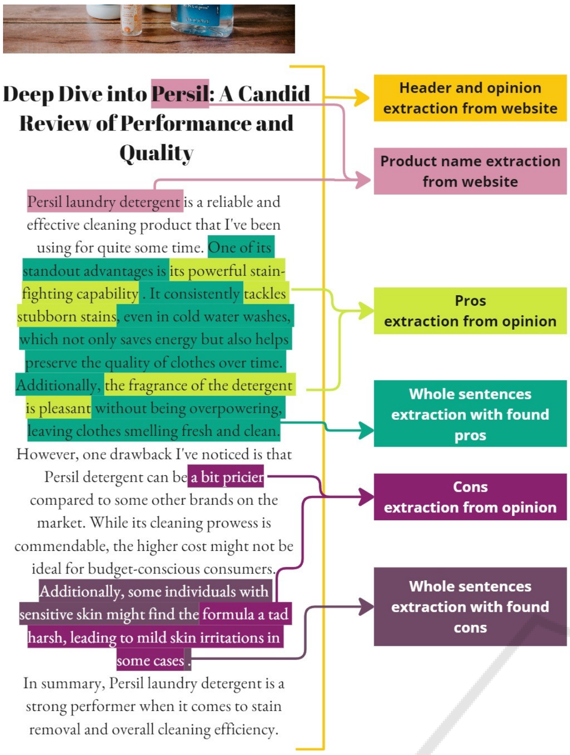
Our web scraping approach:

- Leverages variants of the transformer architecture:
  - BERT (Devlin et al., 2019) for product name

[a] https://orcid.org/0009-0004-7897-0506
[b] https://orcid.org/0009-0004-4388-2880
[c] https://orcid.org/0009-0000-1234-6728
[d] https://orcid.org/0000-0002-1054-4285
[e] https://orcid.org/0000-0002-2442-345X

Plaskowski, D., Skwarek, S., Grajewska, D., Niemir, M. and Ławrynowicz, A.
Automating Opinion Extraction from Semi-Structured Webpages: Leveraging Language Models and Instruction Finetuning on Synthetic Data.
DOI: 10.5220/0012384900003636

In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence (ICAART 2024) - Volume 3*, pages 681-688
ISBN: 978-989-758-680-4; ISSN: 2184-433X



Deep Dive into Persil: A Candid Review of Performance and Quality

Persil laundry detergent is a reliable and effective cleaning product that I've been using for quite some time. One of its standout advantages is its powerful stain-fighting capability . It consistently tackles stubborn stains, even in cold water washes, which not only saves energy but also helps preserve the quality of clothes over time. Additionally, the fragrance of the detergent is pleasant without being overpowering, leaving clothes smelling fresh and clean. However, one drawback I've noticed is that Persil detergent can be a bit pricier compared to some other brands on the market. While its cleaning prowess is commendable, the higher cost might not be ideal for budget-conscious consumers. Additionally, some individuals with sensitive skin might find the formula a tad harsh, leading to mild skin irritations in some cases .

In summary, Persil laundry detergent is a strong performer when it comes to stain removal and overall cleaning efficiency.

Header and opinion extraction from website
Product name extraction from website
Pros extraction from opinion
Whole sentences extraction with found pros
Cons extraction from opinion
Whole sentences extraction with found cons

Figure 1: Functionalities.

classification.

  - FLAN-T5 (Chung et al., 2022) finetuned on instruction-following data and synthetic web-blogs generated with GPT-3.5, GPT-4 to extract product names with their associated pros and cons (Møller et al., 2023).

- Identifies and isolates text chunks relevant to a single product.
- Provides sentences that carry an opinion in each of the chunks.
- Details which part of the sentence signals sentiment related to the product.

The method provides a holistic view of the website, automatically labeling key sentiment aspects at the sentence level, such as the explicitly stated pros and cons of a product. Furthermore, the collected data can be a resource for a variety of machine learning models, depending on the degree of detail required. For example, broader context data can be used to provide an overview of general sentiment trends across multiple reviews, while more granular data can help train models for aspect-based sentiment analysis.
To the best of our knowledge, this work is the first to:

- combine BERT-based text classification of HTML elements with FLAN-T5 for opinion extraction,
- use synthetic data for intruction finetuning of encoder-decoder transformer for the task of information extraction of semi-structured web pages,
- generate instruction data for the e-commerce domain.

The approach was evaluated in the case of web scraping blogger reviews on three product categories: laundry detergents, dishwasher tablets, and insulated cups. This involves processing a total of 100 websites and collecting 900 reviews as a part of train set. The primary focus lies in categorizing HTML elements on web blogs through text classification to link product names with their corresponding reviews. Subsequently, the positive and negative aspects of each product from the reviews were extracted.

2 RELATED WORK

Recent advancements in machine learning-based web scrapers have explored the use of language models to deal with the unstructured text domain of webpages (Li et al., 2022; Wang et al., 2022). By integrating HTML into the training process, these methods aim to develop models that can efficiently extract information from various website structures for tasks such as answering questions and comprehension of the text in SWDE and WebSRC datasets (Hao, 2011; Chen et al., 2021). However, they often fall short in handling the real-world diversity and disorganization found in blogs, being restricted primarily due to their limited context window. There was also created a framework (Xie et al., 2021) known as WebKE that approach utilizes pre-trained language models to extract knowledge from semi-structured webpages by incorporating markup language and encoding layout semantics. In order to enhance the efficient web exploration of emotional content (like opinions) (Vural et al., 2014) was developed a sentiment-oriented web crawling framework. The main objective of this tool is to prioritize the systematic exploration of sentimental content compared to non-sentimental content. In the context of this framework, they present three approaches:a technique on reffering anchor text, a technique based on reffering pages content and technique utilizing machine learning methods.

3 OUR APPROACH

In order to solve the problem of chunk extraction, we first deploy the model which is responsible for extraction of structured data from webpages. To do so, at the beggining the module named ”Proheader”, which is responsible for the extraction of headers and related descriptions was developed. The ”Proheader” module depends on the ”Title Classifier”, which is based on the BERT architecture. (Devlin et al., 2019) We trained our classifier using our own database. The used data for the fine-tune classifier was 2,000 product names, and 10% of them are noisy by joining other words to the front and back - to simulate a title. Another 2,000 examples of negative classes came from website headlines that are not related to products. The diagram 2 presents the process of extraction opinions from a single webpage. The first flow at the top of the diagram shows how we identify the text that reviews the product from the website. The second flow shows how we indicate sentences containing pros, cons, and the product name from the opinion text detected in the previous step.

The main concept behind the idea of ”Proheader Module” uses the fact that in our product domain webpages (blogposts, rankings) it happens very often that product name is hidden inside of a header (one of h1 - h5 html tags). We want to increase a probability that a given header is selected. Therefore, for each header, we produce a list of alternative headers, which contain the whole header’s text and smaller in length parts of a given header - let us name all of those as alternative headers. Then, for each alternative header, we ask our ”Title Classifier” whether it is a product name. If so, we consider the given header as the potential product header.

We define a ”pattern sequence” as a sequence of tag names between two potential product headers with their names included. For example, if a leading and ending headers are h2 tags a pattern sequence might look like: (h2, a, p, p, h3, p, h2), where a and p are hyperlink and paragraph tag names respectively. Let us assume that headers are adjacent if there is no header of the same kind following the pattern sequence between them. Using previously found potential product headers, we find all pairs of adjacent product headers. We define pattern length as the distance between the first and the last of two adjacent headers. We select the most common pattern sequence from sequences found between adjacent headers, preferring the one that has the least length.

We define a list of candidate headers and descriptions as an empty list. The discovered sequence contains a starting header and an ending one. From each of those headers we search for the other one of the same kind, following pattern sequence in both up and down directions. If a header of the same kind is found after performing a given direction, we append it to the list of candidate headers. If the performed search direction was up, the tags on the path to the newly discovered header are directly mapped to its description. Otherwise, we check whether we can perform the pattern sequence once again. If so, found tags are mapped to the header description.

The above procedure produces candidate headers and descriptions. The goal of the procedure was to enhance a list of potential product headers and extract the descriptions between them.

Within the context of extraction of product characteristics module, we identify: product name, pros and cons of a product.

The product names appear in both the headers and the descriptions. We ask the FLAN-T5 model to extract the product name from a header. Then we again ask for the name of the product from a description. We select the best of both names with the use of a ”Title Classifier”. The pros and cons of a product are hidden in its description. We extract them again with the use of the FLAN-T5 model.

In extraction of opinions module it is assumed that a single opinion as a whole sentence about a given product. The characteristics that describe a given product are pros and cons. To address the problem of extraction of entire sentences, we use the information retrieval method ”Okapi BM25” (Amati, 2009), which, given a small text chunk, is capable of finding the most relevant sentence.

4 IMPROVING THE INSTRUCTION TUNED LANGUAGE MODEL

In order to improve our FLAN-T5 model, we fine-tune it with data which we generate using GPT models developed by OpenAI (Brown et al., 2020; OpenAI, 2023).

4.1 Generation of Synthetic Datasets

Part of our research involves the exploration of how the capabilities of large language models such as GPT-3.5 and GPT-4 can be transferred to smaller, more accessible models. An exemplary model in this context is FLAN-T5, which has been pre-trained and fine-tuned on instructions—brief descriptions of tasks that have empirically demonstrated effective performance on out-of-distribution tasks in zero-shot settings.

With a focus on transfer learning, we formulated a two-step approach to synthetic dataset generation for fine-tuning FLAN-T5.

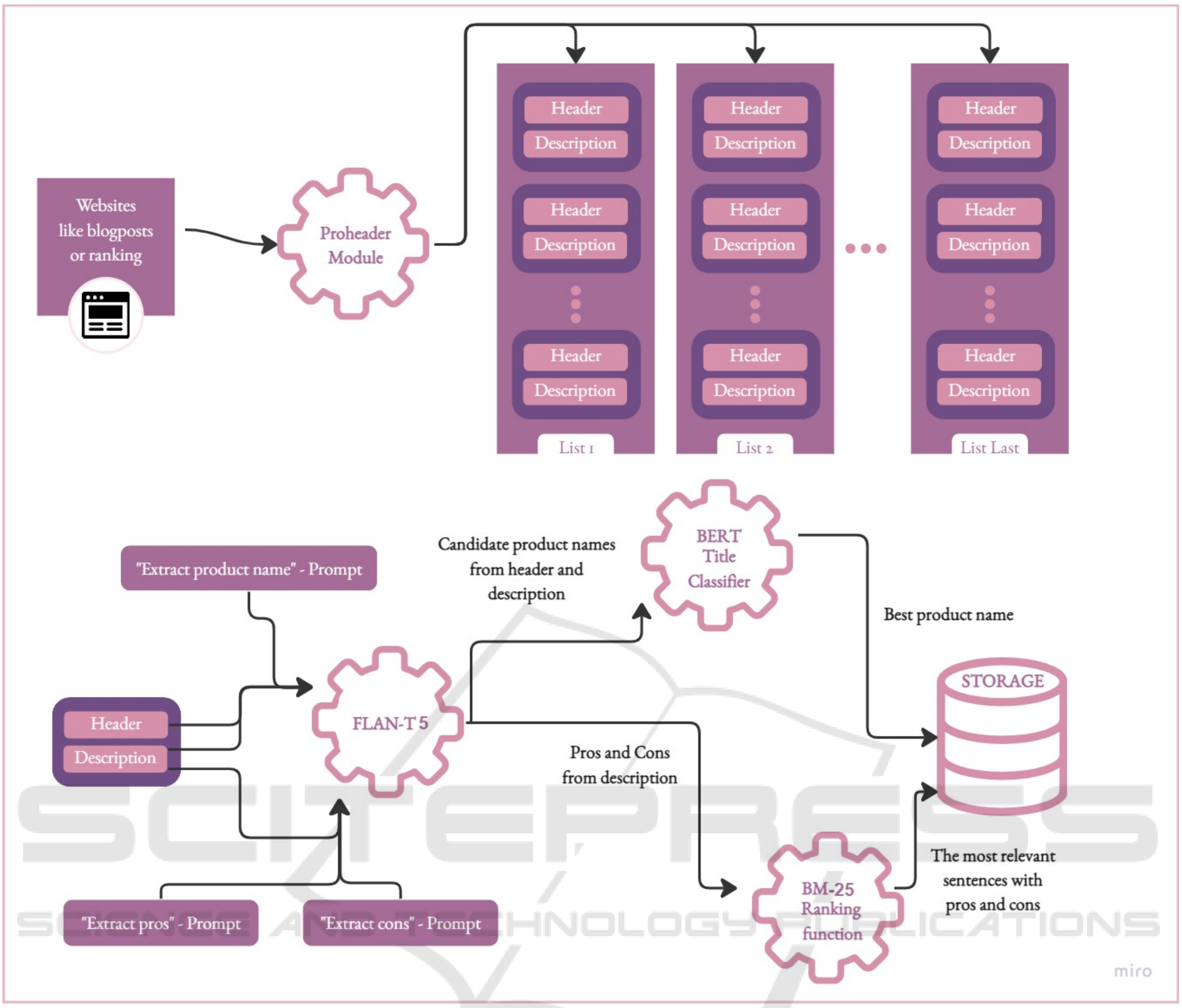


Figure 2: Skeleton of our approach.

1. **Generating Product Domain-Specific Instructions - Instruct Dataset:** The first step involves the creation of instructions tailored to the product domain. The goal here is to adapt FLAN-T5 to understand and process data specific to product reviews, comparisons, and rankings.
2. **Generating Weblog Pages with Varied Styles and Content:** The second step focuses on producing synthetic weblog pages that reflect a diverse distribution of styles (Chen et al., 2022) and content, including specific pros and cons associated with various products. This diversity is vital in replicating the complexity and heterogeneity of real-world blogs, thereby enhancing the model's ability to handle various formats and layouts.

Figure 3 shows the schema of our method to generate synthetic data.
We design prompts that we send to OpenAI's API to produce our datasets.

4.1.1 Instruct Dataset

Following prior work on *instruct finetuning* (Taori et al., 2023; Wang et al., 2023), we generate a dataset which contains product domain instructions. As a result, we obtain 4,606 instructions. A sample general instruction is ”What are the most notable features of the X product line?”, where X is instantiated to a specific product name.

4.1.2 Synth Dataset

We constructed a synthetic dataset of product reviews for the following purposes:

- **Diversity and Control:** Using the GPT-3.5 and GPT-4 models, we generated blog posts about products, incorporating specific styles, personas, problems, and bloggers. Customization, detailed in table 1, allows for a controlled diversification

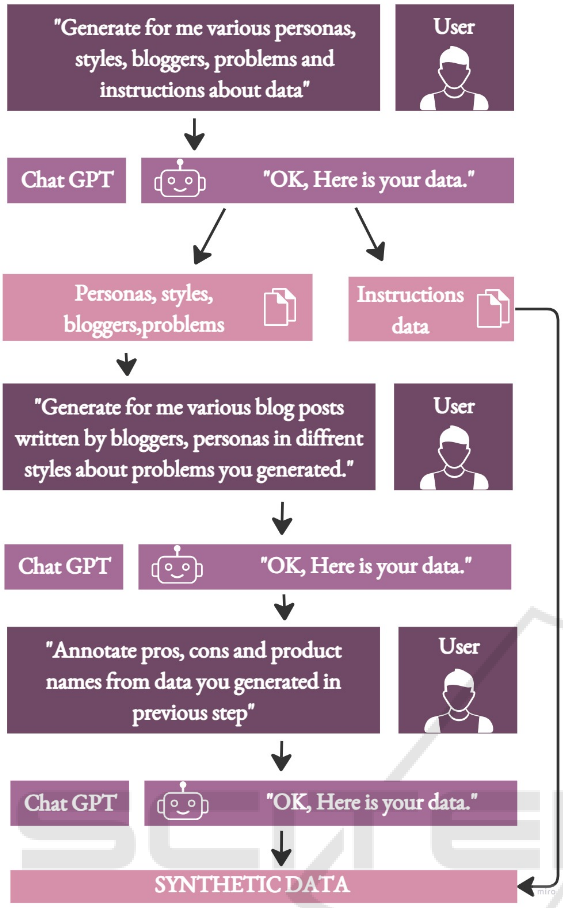

"Generate for me various personas, styles, bloggers, problems and instructions about data"
User
Chat GPT
"OK, Here is your data."
Personas, styles, bloggers,problems
Instructions data
"Generate for me various blog posts written by bloggers, personas in diffrent styles about problems you generated."
User
Chat GPT
"OK, Here is your data."
"Annotate pros, cons and product names from data you generated in previous step"
User
Chat GPT
"OK, Here is your data."
SYNTHETIC DATA

Figure 3: Synthetic data generation schema.

of opinions and styles, reflecting the complexity of real-world blogs.

- We prompt the GPT-3.5, GPT-4 models to generate blogposts about products. The product names are taken from our own database. We enhance the instruction with an order to generate it in a given style and like it was written by a given persona. The 6 personas, 5 problems and different styles and bloggers are generated by GPT. We present our prompts and variable prompts components in table 1. The process of adding styles, personas, problems, and bloggers increases the diversification of opinions.
- To obtain webpages without opinions, we order GPT to generate it without any pros and cons. Similarly, we obtain webpages without product names.
- Then, we ask the GPT to extract product names, pros and cons from generated blogposts.
- **Balancing the Dataset:** To create a more balanced distribution of possible webpages that our model may encounter, we used GPT-3.5 and GPT-4 to generate webpage variations. These variations included some pages devoid of opinions (such as pros and cons) and others without product names. This approach ensured a more diverse and representative set of data, allowing our model to handle text chunks that may not contain pros and cons.

The GPT is ordered to return a valid json string. If it is not valid, we repeat asking the GPT. We used the following prompts:

- ”Let’s assume that {{product_name}} is a hypothetical {{product_type}}. {{actor_action}} this washing powder, its advantages, disadvantages, own and others experiences and opinions. Get all pros span from this text. Get all cons span from this text. Put message only in the JSON structure {”text”:”..”, ”pros_span”:[”..”], ”cons_span”:[”..”]} without any comments.”
- ”Generate some hypothetical washing powder. It should have the product name, including brand and variant. Describe this washing powder, its advantages, disadvantages, own and others experiences and opinions using keywords: {{Problems}}. Get all pros span from this text. Get all cons span from this text. Put message only in the JSON structure {”product_name”:”..”,”text”:”..”, ”pros_span”:[”..”], ”cons_span”:[”..”]} without any comments.”
- Generate blog post about {{PRODUCT_NAME}}. The text must not contain positive and negative features, advantages, disadvantages, pros and cons. This blog should have the following style: {{STYLES}}. This blog should be generated, as if it was written by {{BLOGGER}}.

We obtain a dataset consisting of 15,831 blog posts. There are annotated product names, pros and cons. Similarly, we develop a dataset with titles of webpages - with and without product names. Then, we ask the GPT to extract product names from the titles. The resulting dataset consists of 5,829 titles.

In total, we get a dataset of 21,659 annotated texts, giving us 64,977 instructions. We will use them to fine-tune our model. In table 2 we provide a number of cases where the answer is not presented in a text of a given kind. We are observing an imbalance, as it is much more common for blogs to have pluses than minuses. If we consider percentages of duplicates in answers made by GPT we can observe that:

- in generated texts there are 0.69% of duplicates

Table 1: Bloggers, Problems, Personas and Styles.

Bloggers	Problems	Personas	Styles
Penelope Poeticus	Packaging cuts hands	Blogger, product specialist	Informative
Zenith Masters	Bad choice	Blogger - influencer	Historical
Chuckle McBubbles	Smears	Blogger - mother of young children	Scientific
Professor Laundrylore	Hard to open	Blogger, who enjoys sensationalism	Narrative
Dr. Detergento	Not suitable for...	Blogger - teenager	Satirical
Mythos Mytherson		Blogger writes about their experiences	Expository

Table 2: Count of cases when answer is not found in Synth.

	Count / percent of texts without:		
	Pros	Cons	Product Name
blogs	5310 / 34%	8086 / 51%	8387 / 53%
titles	all	all	2940 / 50%
all	11139 / 51%	13915 / 64%	11327 / 52%

- in pros there are 0.26% of duplicates
- in cons there are 2.53% of duplicates

Therefore, we conclude that our synthesized dataset is diversificated with less than 1% of repetitions in both generated webpages and titles.

4.2 Our Test Set

We run the proheader module on a dataset of 155 candidate webpages. The module produces product lists on 77 out of them. This gives us in total 908 pairs of headers and descriptions.

In case of headers dataset we annotate each of them via marking the span which contains the product name. In only 76 cases there is no product name in the title.

When we consider descriptions dataset, we mark the spans corresponding to: product names, pros of the product and cons of the product. There are 411 descriptions that do not contain any name of a product, 338 which do not contain a disadvantage, and 253 that do not contain an advantage.

We credit Doccano (Nakayama et al., 2018) ”open source” annotation tool, which we use during the span marking process.

5 EXPERIMENTS

During the experiments, our goal is to examine the performance of models of different sizes and fine-tuned on different data, and then compare them. The performance of the models is measured with metrics to evaluate the quality of text prediction (Exact Match, BLEU(Papineni et al., 2002), ROUGE(Lin, 2004)). In the following steps, we examine how the model behaves in cases where the text does not contain any of the information we are looking for. Also, we check the performance of the model by looking at whole sentences that contain the information being searched for.

Table 3: Pros - models comparison.

model	EM	BLEU	ROUGEL-f1
base pure	0.03	0.1	0.1
large pure	0.2	0.23	0.23
base instr.	0.0	0.21	0.25
large instr.	0.01	0.26	0.34
base instr. then p.	**0.25**	0.51	0.58
large instr. then p.	0.23	0.59	0.51
base instr. & p.	0.24	0.51	0.59
large instr. & p.	0.0	0.21	0.25
base p.	0.24	**0.7**	**0.7**
large p.	0.24	0.51	0.59

To compare results, we have tested 10 different Flan-T5 (Chung et al., 2022) models with different sizes and fine-tuned on the basis of various data:

- base and large Flan-T5 without fine-tuning (base pure / large pure),
- base and large Flan-T5 fine-tuned on instructional data (base instruct / large instruct),
- base and large Flan-T5 fine-tuned on data related to blog reviews (base products / large products),
- base and large Flan-T5 fine-tuned first on instructional data and then on data related to blog reviews (base instr. then p. / large instr. then p.),
- base and large fine-tuned simultaneously on instructional data and product reviews on blogs (base instr. & p. / large instr. & p.)

The round brackets contain the abbreviated names of the models under which they appear in the following tables.

5.1 Evaluation

The main goal of the experiment is to evaluate our approach and compare the models in how they deal with the problems we have prepared: extracting the

Table 4: Cons - models comparison.

model	EM	BLEU	ROUGEL-f1
base pure	0.02	0.19	0.14
large pure.	0.46	0.6	0.55
base instr.	0	0.11	0.15
large instr.	0.21	0.51	0.46
base instr. then p.	0.65	0.76	0.79
large instr. then p.	0.65	0.76	0.79
base instr. & p.	0.63	0.75	0.78
large instr. & p.	0.21	0.51	0.46
base p.	0.63	0.75	0.78
large p.	**0.66**	**0.77**	**0.80**

Table 5: Product names - models comparison.

model	EM	BLEU	ROUGEL-f1
base pure	0.36	0.5	0.56
large pure	0.53	0.6	0.63
base instr.	0.4	0.54	0.58
large instr.	0.63	0.71	0.72
base instr. then p.	0.59	0.69	0.71
large instr. then p.	0.66	0.76	0.74
base instr. & p.	0.59	0.7	0.72
large instr. & p.	0.63	0.71	0.72
base p.	0.61	0.72	0.71
large p.	**0.65**	**0.73**	**0.75**

name, pros and cons from the product review text. Tables 3 to 5 show a comparison of model results for three different tasks: pros, cons, and product names extraction using Exact Match (EM), BLEU and ROUGEL-F1 metric.

After testing all variants of the model, it was concluded that the large Flan-T5 model, which was fine-tuned only on product-related data, performed best on the task of extracting product names and cons from product opinions (as can be seen from the Exact Match, BLEU, and ROUGEL-F1 metrics in Tables 4 and 5).

During the task of extracting product pros in the BLEU and ROUGEL-F1 metrics, the base model fine-tuned only on product-related data performed best. However, the base model fine-tuned first on instructions and then on product-related data has the highest Extract Match value (Table 3).

The significant differences in metrics for the same models, but for different tasks, is due to the fact that the test set for each task differs in the amount of text which does not contain searched information. The same model may deliver different results in the identification of information or in the simple indication of information. Our data contain real data in original proportions, so it would be inconsistent to artificially balance the set to have the same number of ”ANSWERNOTFOUND” cases. Texts are much more likely to contain product pros, rather than cons. Therefore, when detecting defects, the most probable correct answer will be ”ANSWERNOTFOUND.”

Table 6: Information Retrieval - Pros from opinion.

filename	precision	recall	f1	k
large instr. & p.	**0.44**	**0.61**	**0.5**	**4**
large instr. then p.	0.44	0.61	0.49	4
large p.	0.44	0.61	0.49	4
base instr. & p.	0.44	0.61	0.49	4
base p.	0.44	0.6	0.49	4

Table 7: Information Retrieval - Cons from opinion.

filename	precision	recall	f1	k
base instr. then p.	**0.66**	**0.76**	**0.69**	**4**
large instr. then p.	0.66	0.72	0.68	3
base p.	0.65	0.75	0.68	4
large p.	0.67	0.72	0.68	3
large p.	0.65	0.76	0.68	4

5.2 Information Retrieval Evaluation

In the next experiment, we want to measure the effectiveness of the models in terms of extracting whole sentences that contain the information we seek. We use metrics like precision, recall, and f1, turning the problem into classification (only for evaluation process), whether the expected whole sentence was predicted by the model.

We used the BM25 algorithm to search for whole sentences in opinions that contain pros, cons, and product names indicated by the model. Table 6, 7 and 8 show the individual metrics for different models and the parameter k, which was used in the get_top_n() function in the BM25 algorithm.

6 ETHICAL CONSIDERATIONS

We observed significant qualitative improvements when training FLAN-T5 using synthetic data from

Table 8: Information Retrieval - Product names from opinion.

filename	precision	recall	f1	k
large instr. & p.	**0.56**	**0.68**	**0.6**	**3**
large instr. & p.	0.54	0.72	0.59	4
large instr.	0.56	0.65	0.59	3
large p.	0.55	0.67	0.58	3
large instr.	0.53	0.69	0.58	4

GPT-3.5 and GPT-4. This suggests that distilling knowledge from larger models holds promise for practical language model applications and emphasizes the importance of further analyzing how the synthetic data enhanced our model. However, given the broader implications, we chose not to open-source the synthetic dataset. Our primary concern is the potential for these data to inadvertently contaminate future web scraping efforts, which could compromise the quality of the data for training upcoming language models.

REFERENCES

Amati, G. (2009). *BM25*, pages 257–260. Springer US, Boston, MA.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners.

Chen, S., Neves, L., and Solorio, T. (2022). Style transfer as data augmentation: A case study on named entity recognition. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1827–1841, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Chen, X., Zhao, Z., Chen, L., Ji, J., Zhang, D., Luo, A., Xiong, Y., and Yu, K. (2021). WebSRC: A dataset for web-based structural reading comprehension. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4173–4185, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Chung, H. W., Le Hou, S. L., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. (2022). Scaling instruction-finetuned language models.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding.

Hao, Q. (2011). Structured web data extraction dataset (swde).

Li, J., Xu, Y., Cui, L., and Wei, F. (2022). MarkupLM: Pre-training of text and markup language for visually rich document understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6078–6087, Dublin, Ireland. Association for Computational Linguistics.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Møller, A. G., Dalsgaard, J. A., Pera, A., and Aiello, L. M. (2023). Is a prompt and a few samples all you need? using gpt-4 for data augmentation in low-resource classification tasks.

Myers, D. and McGuffee, J. W. (2015). Choosing scrapy. *J. Comput. Sci. Coll.*, 31(1):83–89.

Nakayama, H., Kubo, T., Kamura, J., Taniguchi, Y., and Liang, X. (2018). doccano: Text annotation tool for human. Software available from https://github.com/doccano/doccano.

OpenAI (2023). Gpt-4 technical report.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Sharma, M. (2014). Selenium tool: A web based automation testing framework.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. (2023). Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Vural, A. G., Cambazoglu, B. B., and Karagoz, P. (2014). Sentiment-focused web crawling. *ACM Trans. Web*, 8(4).

Wang, Q., Fang, Y., Ravula, A., Feng, F., Quan, X., and Liu, D. (2022). Webformer: The web-page transformer for structure information extraction. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 3124–3133, New York, NY, USA. Association for Computing Machinery.

Wang, Y., Kordi, Y., Mishra, S., Liu, A., Smith, N. A., Khashabi, D., and Hajishirzi, H. (2023). Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Xie, C., Huang, W., Liang, J., Huang, C., and Xiao, Y. (2021). Webke: Knowledge extraction from semi-structured web with pre-trained markup language model. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, CIKM '21, page 2211–2220, New York, NY, USA. Association for Computing Machinery.