

Smart Surveys: An Automatic Survey Generation and Analysis Tool

Augusto Gonzalez Bonorino ^a

Claremont Graduate University, Claremont, California, U.S.A.

Keywords: Survey Methodology, Artificial Intelligence, Computer Architecture, Natural Language Processing, Large Language Models.

Abstract: This position paper presents a proof of concept for a novel architecture designed to automate the process of survey design and analysis. The architecture leverages the power of Python automation, Artificial Intelligence (AI), and Large Language Models (LLMs) to streamline the survey process and provide actionable insights. The architecture includes various key components such as data generation, survey distribution, data collection, preprocessing, text mining, and report generation. Through two use cases in the field of education, the paper demonstrates how the architecture can empower students and instructors to conduct accurate research and make informed decisions. The use cases illustrate how Smart Surveys can reduce development time, flatten the learning curve, and provide actionable insights via interactive visualizations and results from AI-generated prompt-based tasks.

1 INTRODUCTION

Surveys are a valuable tool for collecting data and gaining insights into various industries, such as education, marketing, healthcare, and social sciences. However, traditional survey analysis methods can be time-consuming and resource-intensive (de Leeuw, 2013), and may not provide as comprehensive or accurate insights as desired. The adoption of electronic surveys, accelerated during the COVID-19 pandemic, has facilitated the automation and maintenance of survey technology at scale. Moreover, Artificial intelligence (AI) and Large Language Models (LLMs) have the potential to improve electronic survey data analysis by automating manual tasks and extracting custom insights from natural language data.


The purpose of this paper is to introduce an architecture for automatic survey generation and analysis that utilizes traditional text mining techniques and large language models to improve survey data analysis in a custom domain. Smart Surveys, is implemented as a Python package, that comprises three main parts. The first part, data generation, uses the large language model to automatically generate relevant survey questions or augment the dataset with synthetic data from alternative sources such as social media. The second part, survey distribution and data collection, provides functionalities to distribute

and collect the surveys via open-source APIs. Lastly, the insights generation part, uses the text processing modules to preprocess the text, mine text characteristics, generate smart insights, and create a comprehensive report customized via prompts. The program is designed to be flexible and modular, allowing users to easily integrate and customize the various components to fit their specific needs. The architecture is designed to empower students, researchers, and educators to conduct accurate research and make informed decisions with minimal technical requirements, reducing development time and flattening the learning curve.

Specifically, the objectives of this paper are to:

1. Describe the architecture, including the use of large language models and multiple modules for text processing.
2. Outline the steps in the pipeline, including generating surveys, distributing and collecting surveys, preprocessing text, mining text insights, and generating smart insights and reports.
3. Provide specific examples of potential use cases for the technology in the Education industry.
4. Discuss the advantages and disadvantages of AI, and LLMs in particular, as a tool to advance survey methodology.

The paper is divided into four sections: a review of existing literature on AI in survey development

^a  <https://orcid.org/0000-0002-9355-0831>

and analysis, an explanation of the architecture and components used, an examination of potential applications of Smart Surveys in Education, and a conclusion that summarizes the benefits and limitations of AI-based survey analysis methods and suggests future research directions.

2 LITERATURE REVIEW

Over the last four decades survey methods have evolved rapidly from face-to-face to paper, then phone, and currently digital surveying approaches; particularly in Higher Education (Moss and Hendry, 2002). The shift has primarily been stimulated by a combination of rising costs of traditional methods and technological innovation. De Leeuw has reviewed survey methods published on BMS issues since 1983 (de Leeuw, 2013). Since de Leeuw's publication, Machine Learning (ML) techniques have been increasingly implemented by survey researchers in a variety of ways. Buskirk et al. (Buskirk et al., 2018) describe how ML algorithms such as LASSO regression and Support Vector Machines (SVMs) (Cortes and Vapnik, 1995) are being used in bias adjustment, understanding correlates of nonresponse, tailoring the survey experience, and other prediction or classification tasks. Moreover, in 2020, Saputri and Lee published a systematic review of ML techniques applied to self-adaptive systems (Saputri and Lee, 2020) with useful information that can be extrapolated to adaptive survey design. Finally, one interesting benefit of web-based surveys is the possibility of automatically augmenting the survey data with external data sources or relevant paradata (West, 2011). Statisticians Lohr and Raghunathan explore novel ways of combining information from multiple sources (Lohr and Raghunathan, 2017), as well as their limitations, to improve survey analysis. This line of research has influenced the data collection and sampling methods implemented in the architecture presented in this paper.

In parallel, recent advancements in Natural Language Processing (NLP) (Omar et al., 2022), pre-trained models (Yosinski et al., 2014), and causal Large Language Models (LLMs) (Feder et al., 2020) uncovered new ways of extracting insights from textual data. More importantly, most of these models are open source. This means that language models such as OpenAI's GPT-3 (Brown et al., 2020) and those available in HuggingFace (Wolf et al., 2019) hub can be incorporated into systems to parse survey responses or aid in the questionnaire design process in real-time at a minimal cost. Nevertheless, despite

many of the promising benefits of AI technology, and LLMs in particular, several challenges remain.

Language Models are commonly expanded along three dimensions: model size, dataset size, and/or computational power. Surprisingly, scaling language models passed a threshold triggers unpredictable emergent abilities. That is, abilities that were not present in the smaller versions of the model but emerged after a critical scale was crossed. Amidst the rising popularity of GPT-like models, Wei et al. define the concept of emergent abilities of LLMs and explore methods to study and evaluate such emergence (Wei et al., 2022). But, with emergent benefits, we should also expect emergent risks. Ethical bias, privacy, trustworthiness, explainability, and toxicity are pressing issues in Artificial Intelligence. Weidinger et al. propose (Weidinger et al., 2021) six risk areas to foster responsible innovation across industries. Finally, if the quality of emergence continues to hold at new record scales in the coming years there exists the possibility that unknown, unpredictable, behaviors may emerge. Hendrycks et al. identify four "emerging safety challenges" in machine learning (Hendrycks et al., 2021): robustness, monitoring, alignment, and systemic safety.

3 ARCHITECTURE DESIGN

The architecture for automatic survey generation and analysis uses the Python programming language to implement a large language model and multiple modules for text processing, question generation, recommendations for improving the survey questionnaire, generating custom prompt-based insights, and a full report that aggregates the various insights generated automatically. It consists of 3 main parts, illustrated in Figure 1.

The first part of the Smart Surveys architecture, "Question and Data Generation", enables researchers to efficiently generate relevant survey questions based on a given set of prompts or topics, or upload existing questions from a database or file. This functionality can be particularly beneficial for researchers who have limited experience with survey methodology or software, as it allows them to quickly and easily generate a set of appropriate and relevant survey questions. Additionally, the user has the flexibility to edit and customize the generated or uploaded questions to ensure they align with the specific research goals and industry requirements. The edited and customized questions are then stored in a database for easy access and further use throughout the survey design and analysis process.

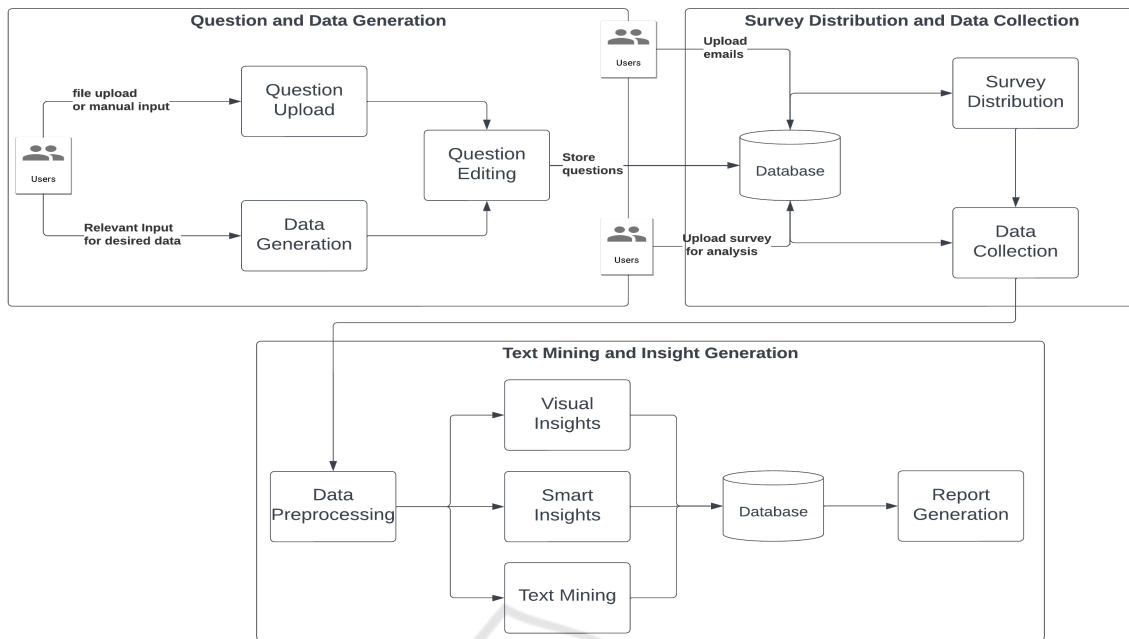


Figure 1: Smart Surveys Architecture.

The second part of the architecture, "Survey Distribution and Data Collection", is currently under development. This component will facilitate the distribution of the generated and edited questions through various channels such as email, social media, or a link. It will also allow for integration with popular survey distribution APIs or accept an input file containing a list of emails for survey distribution. The ability to integrate with popular survey software, using their python APIs, provides centralized control and streamlined data collection from multiple sources. Additionally, it will include features such as automated response tracking, enabling users to efficiently monitor and analyze survey progress. This design promotes broader utility by enabling integration with commonly used survey software. However, this is an optional component, as users can choose to manage their survey distribution independently and simply leverage the insights functionalities provided by the architecture.

The third part, "Text Mining and Insight Generation", focuses on cleaning and preprocessing collected survey data to remove bias and grammatical errors, and enhance the accuracy of the text mining process. The user can select from a range of text mining techniques, such as sentiment analysis, named entity recognition, and topic modeling, to extract insights from the cleaned data. The Smart Surveys architecture also enables users to use a large language model to generate custom insights or utilize the default features provided. To aid in analysis, most features are

enhanced with visualization methods, such as interactive bigram networks and keyword count histograms. A full report is generated to aggregate survey data and insights for a comprehensive understanding of the survey data.

The technology stack used to develop the Smart Surveys architecture leverages stable, open-source resources that are easily scalable and maintainable with minimal additional cost. The use of open-source models promotes transparency and fosters collaboration and community-driven development, leading to faster innovation and a wider range of applications for the architecture. Furthermore, this allows institutions to build on top of the baseline architecture of Smart Surveys to efficiently meet their specific needs. Its modular design allows for updating or replacing components without affecting the overall functioning of the program. Table 1 lists the technical components and their functionalities used in Smart Surveys.

4 APPLICATIONS

This section illustrates the capabilities and effectiveness of the proposed architecture through two use cases: Student Research and Course Evaluation. These use cases were chosen as they represent common scenarios in which survey data is used to gain insights and make informed decisions in Education. The first use case, Student Research, demonstrates how the Smart Surveys architecture can empower un-

Table 1: Programming modules implemented in Smart Surveys.

Technology	Version	Description
Numpy (Oliphant, 2006)	1.24.1	Array manipulation library for numerical computation in Python.
OpenAI (Openai, 2020)	0.26.0	Access GPT-like models for language processing tasks.
Transformers (Wolf et al., 2020)	4.25	Python module for loading publicly available models and fine-tuning them.
Pandas (Wes McKinney, 2010)	1.5.2	Data manipulation and analysis library for handling large datasets.
Requests (Foundation, 2011)	2.28.2	Library for making HTTP requests in Python.
NLTK (Bird et al., 2009)	3.8.1	Natural language processing library for text preprocessing, tokenization and other language-related tasks.
Rake-nltk (Rose et al., 2010)	1.0.6	Library for extracting key phrases in a body of text using the RAKE algorithm.
Scikit-learn (Pedregosa et al., 2011)	1.2.0	Access machine learning models for text mining tasks.
TextBlob (Sloria, 2013)	0.16	Natural language processing library for tasks such as part-of-speech tagging, noun phrase extraction and sentiment analysis.
Json (Van Rossum and Drake, 2009)	3.11.1	Data storage method for survey data, allows for easy modification and integration with other platforms.
Pyvis (Perrone et al., 2020)	0.3.1	Library for creating interactive network diagrams.
Matplotlib (Hunter, 2007)	3.6.3	Data visualization library for creating different types of plots and charts.
Spellchecker (Barrus, 2018)	0.4	Library for spell checking and language processing.
re (Van Rossum and Drake, 2009)	3.11	Built-in python library for working with regular expressions.

dergraduate students who are not familiar with survey methodology or software to conduct accurate research with minimal technical requirements; flattening the learning curve and reducing development time. The second use case, Course Evaluation, illustrates how the architecture can be used to improve the effectiveness of course evaluations and provide actionable insights to course instructors and academic departments from anonymous surveys with the aid of AI systems.

4.1 Student Research

Consider an undergraduate student who has decided to conduct a research project on the study habits of his peers before and after the move to hybrid education during the COVID-19 pandemic. The student has limited experience with survey methodology and software, and must complete the project within one semester. Therefore, he/she has approximately 4 months to choose the technology stack, become fa-

miliar with it, design the experiment, learn to write good questions, manipulate the data, conduct statistical analyses, and write a report on the survey data with meaningful insights. This example shows how Smart Surveys reduces the number of tasks by automating various components of survey design. In turn, students can allocate more time on the experiment design and critical thinking activities such as interpreting insights, trends, or visualizations.

Initially, the student provides a brief description of the study, including the main goal and purpose, as well as a few sample questions for reference and any other relevant details. This information is used by the pipeline's **Data Generation** component to automatically generate a set of relevant survey questions. The student then has the opportunity to review and edit the generated questions before they are stored in a local JSON database for later use.

Next, the student uses the pipeline's **Survey Distribution** component to take advantage of the integra-

tion with Google Forms to easily create the survey, set deadlines, and distribute it to participants via a link that can be shared via email or social media. As survey responses are submitted, the pipeline's **Data Collection** component automatically collects and stores the survey data for preprocessing.

The **Data Preprocessing** component corrects grammatical errors, removes any irrelevant or biased data, and performs other optional normalization techniques for both text and numerical data. Cleaning the data is essential for the accuracy of the insights generated later on; yet, modern techniques are rarely taught outside computational statistics-oriented majors. Thus, automating this step will allow students from all majors to implement a variety of methods without any programming knowledge required. The collected and preprocessed survey data is then fed to the **Text Mining** and **Smart Insights** components for further analysis.

For the project's purposes, no advanced analysis technique is needed, so the student uses the **Text Mining** component to extract the ten most frequent keywords and compute the sentiment analysis of each textual response. These techniques help to identify patterns and trends in the data that would be difficult to discern from manual analysis. One of the most powerful features of the Smart Surveys architecture is the ability to visualize the data in an interactive and easy to understand format. The visualization features available via the **Visual Insights** include interactive network graphs of bigrams, word count histograms, polarity comparison charts, and other types of plots that aid in the analysis of survey results.

Before generating the full report, the student leverages the **Smart Insights** component default functionalities to identify the main patterns or phrases in the textual responses, accounting for each response sentiment scores and extracted keywords. Alternatively, he/she can supplement the baseline prompt with a custom instruction to generate insights that are tailored to the specific needs of their research. Finally, the **Report Generation** component generates a full report by aggregating the survey data and all the insights generated. The report gives the user a comprehensive understanding of the survey data and extracts insights while keeping the intended audience in mind. For instance, it is common to produce reports of different granularity for the different departments, within a company, involved in the research.

In summary, Smart Surveys provides an easy and efficient solution for students conducting research with survey data. By automating the main components of survey design and analysis, Smart Surveys reduces development time and encourages more stu-

dents to use surveys in their research. Additionally, the architecture is designed to be easy to use and understand, allowing students to learn about survey methodology as they progress through their project.

4.2 Course Evaluation

The second use case for Smart Surveys is for course evaluations. The goal of this use case is to demonstrate how the architecture can be used in a more traditional setting, where the user already has a set of survey questions, but wants to leverage the AI features of Smart Surveys to improve the survey results and generate a report automatically for various audiences.

In this scenario, a professor at a university wants to evaluate the effectiveness of a course he/she is teaching. The user already has a set of survey questions that have been used in the past but wants to make sure that the questions are still relevant and effective. To do this, the survey questions are uploaded to the **Question Upload** component of Smart Surveys.

Once the questions are uploaded, the professor has the option to access the AI features of Smart Surveys, such as paraphrasing or generating similar questions, and can edit the questions to ensure they are relevant and appropriate for the course evaluation. Once the questions have been edited, they are stored in the database for future use. The professor then decides to download the questions and use personal software to distribute the survey to students.

As the students complete the survey, the responses are uploaded to the database for the **Data Collection** component to fetch. Then, the responses are preprocessed and fed as input to the selected insight generation functions, where the data is cleaned, normalized, and analyzed using various text mining techniques such as sentiment analysis, named entity recognition, and topic modeling. This is done while accounting for the anonymity of survey responses, which limits the granularity to which we can access information.

Finally, the insights generated are aggregated in one temporary database optimized for rapid data extraction. The **Report Generation** component fetches all of the information and generates a comprehensive report of the entire survey, including the insights discovered previously, automatically using a Large Language Model. This report can be customized via prompts.

The course evaluation use case demonstrates how Smart Surveys can be used in a traditional setting, while still leveraging AI features to improve the survey results. The flexibility of the architecture allows the user to use their own method of survey distribu-

tion, while still being able to access the powerful AI features of Smart Surveys.

5 CONCLUSION AND FUTURE RESEARCH

In this position paper, I proposed the Smart Surveys architecture, a pipeline for the automated generation, distribution, and analysis of survey data. The architecture is built using the Python programming language and leverages a large language model and multiple modules for text processing, question generation, recommendations for improving the survey questionnaire, generating custom prompt-based insights, and a full report that aggregates the various insights generated automatically. Applying Smart Surveys in two selected use cases, it has been demonstrated how it can empower undergraduate students and researchers to conduct accurate research with minimal technical requirements, and provide actionable insights to course instructors and academic departments with the aid of AI systems.

One of the main benefits of the Smart Surveys architecture is its scalability. It is built using Python and open-source libraries, making it easy to adapt to different industries and research goals, and it can be easily scaled to handle a large number of survey responses. However, it is important to note that the quality of the generated survey questions and insights depends on the quality of the data and the information provided by the user. Thus, it will be important to incorporate educational material for the users on best practices when communicating with LLMs.

Looking forward, there are several areas of future research that could further enhance the capabilities of the Smart Surveys architecture. One avenue is to improve the baseline features of the pipeline to incorporate more robust survey design methods and sampling techniques to improve the AI's suggestions and capabilities. Another important area for future research is to further investigate the ethical implications of using AI in survey research, such as privacy, bias, and explainability. With the increasing use of AI in survey research, it is crucial to ensure that the data collected is protected and that the AI systems used do not perpetuate biases and discrimination. Furthermore, it is important to explore ways to make AI systems more transparent and explainable, allowing for greater trust and understanding of the results generated. This can be achieved by incorporating techniques such as counterfactual analysis, which can help to identify and understand the factors that influence the AI's predictions and decisions, and by

providing more detailed explanations of the AI's reasoning behind its insights and recommendations. I hope this research encourages researchers to explore other avenues in which AI can help enhance survey data, a valuable resource for all social sciences.

REFERENCES

- Barrus, T. (2018). Pure python spell checking. <https://github.com/barrust/pyspellchecker>. [Online; accessed March 1, 2023].
- Bird, S., Klein, E., and Loper, E. (2009). *Natural language processing with python*. O'Reilly Media Inc.
- 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. J., 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. *ArXiv*, abs/2005.14165.
- Buskirk, T., Kirchner, A., Eck, A., and Signorino, C. S. (2018). An introduction to machine learning methods for survey researchers. *Survey practice*, 11:2718.
- Cortes, C. and Vapnik, V. N. (1995). Support-vector networks. *Machine Learning*, 20:273–297.
- de Leeuw, E. D. (2013). Thirty years of survey methodology / thirty years of "bms". *BMS: Bulletin of Sociological Methodology / Bulletin de Méthodologie Sociologique*, (120):47–59.
- Feder, A., Oved, N., Shalit, U., and Reichart, R. (2020). Causalm: Causal model explanation through counterfactual language models. *Computational Linguistics*, 47:333–386.
- Foundation, P. S. (2011). Psf/requests: A simple, yet elegant, http library. <https://github.com/psf/requests>. [Online; accessed March 1, 2023].
- Hendrycks, D., Carlini, N., Schulman, J., and Steinhardt, J. (2021). Unsolved problems in ml safety.
- Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95.
- Lohr, S. L. and Raghunathan, T. E. (2017). Combining Survey Data with Other Data Sources. *Statistical Science*, 32(2):293 – 312.
- Moss, J. and Hendry, G. (2002). Use of electronic surveys in course evaluation. *British Journal of Educational Technology*, 33.
- Oliphant, T. E. (2006). *A guide to NumPy*, volume 1. Trelgol Publishing USA.
- Omar, M., Choi, S., Nyang, D., and Mohaisen, D. (2022). Robust natural language processing: Recent advances, challenges, and future directions. *arXiv preprint*, arXiv:2201.00768.
- Openai (2020). Openai/openai-python: The openai python library provides convenient access to the openai api

- from applications written in the python language. <https://github.com/openai/openai-python>. [Online; accessed March 1, 2023].
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.
- Perrone, G., Unpingco, J., and Lu, H.-m. (2020). Network visualizations with pyvis and visjs. *arXiv preprint*, arXiv:2006.04951.
- Rose, S., Engel, D., Cramer, N., and Cowley, W. (2010). *Automatic Keyword Extraction from Individual Documents*, pages 1 – 20.
- Saputri, T. R. D. and Lee, S.-W. (2020). The application of machine learning in self-adaptive systems: A systematic literature review. *IEEE Access*, 8:205948–205967.
- Sloria (2013). Sloria/textblob: Simple, pythonic, text processing–sentiment analysis, part-of-speech tagging, noun phrase extraction, translation, and more. <https://github.com/slوريا/textblob>. [Online; accessed March 1, 2023].
- Van Rossum, G. and Drake, F. L. (2009). *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.
- Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. (2022). Emergent abilities of large language models.
- Weidinger, L., Mellor, J., Rauh, M., Griffin, C., Uesato, J., Huang, P.-S., Cheng, M., Glaese, M., Balle, B., Kasirzadeh, A., Kenton, Z., Brown, S., Hawkins, W., Stepleton, T., Biles, C., Birhane, A., Haas, J., Rimell, L., Hendricks, L. A., Isaac, W., Legassick, S., Irving, G., and Gabriel, I. (2021). Ethical and social risks of harm from language models.
- Wes McKinney (2010). Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61.
- West, B. T. (2011). Paradata in survey research. *Survey Practice*, 4:1–8.
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface’s transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.
- Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? *arXiv preprint*, arXiv:1411.1792.