Maximizing Productivity and Quality in Senior Thesis Writing with Artificial Intelligence and Natural Language Processing Driven Tools

Lauren Leadbetter

CLAREMONT MCKENNA COLLEGE

Maximizing Productivity and Quality in Senior Thesis Writing with Artificial Intelligence and Natural Language Processing Driven Tools

submitted to

Dr. Michael Izbicki

by

Lauren Leadbetter

for

Senior Thesis

2023

April 24

Abstract

This project is a Python program designed to generate a senior thesis on a user-inputted topic using natural language processing techniques. The program takes in a topic from the user and then uses OpenAI API to deploy text models for text generation and evaluation, such as GPT-3 and Davinci-003. The resulting output is in .tex format and includes a first-draft outline and paper, followed by self-generated assessment, with scoring, revisions, and feedback comments instructing manual revisions.

This submission is a sample using one available model of the project, meant to demonstrate it's functionality and limitations. Further model versions are available in the project repository.

Acknowledgments

Studying Data Science at CMC has taught me invaluable lessons and skills and changed my outlook on the world around me. I would like to thank my family and friends for supporting me, pushing through piles of problem sets, working late at night on impossible projects, and eventually finding my voice and style in code.

I would also like to thank my advisor, Mike, who through years of taking his classes at CMC has helped me form my knowledge of data science and its power and place in the world around us. Without his guidance on this project, it wouldn't have been possible to make something so meaningful, and I wouldn't have gained the invaluable confidence in my abilities that I have today.

Lastly, I want to express my gratitude to the Claremont McKenna College Math Department for working incredibly hard to create a space for my understanding to grow, in class, office hours, and at the QCL. Thank you all so much for bringing interesting, clever, challenging, and rewarding problems to all of your classes. Your teaching has instilled in me a deep appreciation for math and data science that I will cherish always.

Table of Contents

1 Project Description

This project is a Python program designed to generate a senior thesis on a user-inputted topic using natural language processing techniques. The program takes in a topic from the user and then uses OpenAI API to deploy text models for text generation and evaluation, such as GPT-3 and Davinci-003. The resulting output is in .tex format and includes a first-draft outline and paper, followed by self-generated assessment, with scoring, revisions, and feedback comments instructing manual revisions.

The goal of this project is to provide a tool for students and researchers to quickly generate high-quality senior theses on a variety of topics, without having to spend months conducting research and writing. The intended audience for this program is undergraduate students in fields such as computer science, data science, and the social sciences.

The program is built using Python 3 and utilizes OpenAI API for text generation and evaluation. It can be run on any operating system that supports Python, and installation instructions are provided further in the thesis file and on the project's github repository linked at the bottom of this page. OpenAI's API leverages the power of NLP and deep learning without having to implement all of the complex algorithms and techniques from scratch. Instead, you can use the API and model to handle tasks such as language generation, text classification, sentiment analysis, and more, which can save you time and effort.

The program is designed to be user-friendly and includes detailed documentation on how to use it, as well as a sample input and output file to demonstrate its functionality. The code is also well-commented and organized, making it easy for other developers to understand and contribute to the project.

Overall, this project represents a novel approach to senior thesis writing that combines the power of natural language processing with the structure and rigor of academic research. By providing a tool that streamlines the research and writing process, we hope to enable more students to produce high-quality senior theses and contribute to the advancement of knowledge in their respective fields.

1.1 GitHub Repository

A live version of this project will be kept on my GitHub repository linked below.

Link: `https://github.com/laurenleadbetter/senior_thesis_generator`

2 Instructions

To run this program, you will need to install Python 3 and several libraries and packages. We recommend using a virtual environment to keep your dependencies separate from other Python projects you may be working on.

2.1 Dependencies

The following dependencies are required to run this program:

- `openai` (version 0.11.2 or higher)
- `json` (included with Python 3)
- `os` (included with Python 3)
- `datetime` (included with Python 3)

You can install the `openai` library using the following command line in the terminal:

```
pip install openai
```

. Note that you may need to use `sudo` or run the command in a virtual environment depending on your Python setup.

2.2 API Key

To use the OpenAI API, you will need to obtain an API key from OpenAI. You can sign up for an account and obtain an API key on the OpenAI website.

Once you have obtained your API key, create a file named `api_key.py` in the project directory and add the following line:

```
OPENAI_API_KEY = "your-api-key-goes-here"
```

Replace `"your-api-key-goes-here"` with your actual API key.

2.3 Running the Program

To run the program, navigate to the project directory and execute the following command:

```
python3 version_17.py
```

Replace `version_17.py` with your desired version. This will start the program and prompt you for a topic. Once you enter a topic, the program will begin generating your thesis text and feedback. The output will be saved to a .txt file with a timestamped filename. Note that to access the output you will need to specify an output location in the function `write_output_to_file`. Running the program may take some time depending on the length and complexity of the input topic, print statement will signal what function is being completed.

3 Model Versions

To demonstrate how the model works model `version_17.py` is included below, followed by sample outputs. Other model versions are available in the project GitHub repository accompanied by descriptions and differences.

3.1 Code

```python
import time
start_time = time.time()
import openai
from api_key import OPENAI_API_KEY
import os
import json
from datetime import datetime

# Set up OpenAI API key
openai.api_key = OPENAI_API_KEY

####################################################
# Generate outline
####################################################

# Define function to generate outline
def generate_outline(topic):
    # Define prompt
    prompt = f"Please write an outline on {topic}, using roman numerals for section headings and latin letters for sub-headings."

    # Call OpenAI API to generate outline
    response = openai.Completion.create(
        model='text-davinci-003',
        prompt=prompt,
        temperature=0.8,
        max_tokens=3800,
        top_p=1,
        frequency_penalty=0,
        presence_penalty=0.6,
    )

    # Extract and return outline
    outline = response.choices[0].text.strip()
    return outline


#####################################################
# Split outline into sections
#####################################################

#lists less problematic than RegExs
roman_numerals = ("I.", "II.", "III.", "IV.", "V.", "VI.", "VII.", "VIII.", "IX.", "X.", "XI.", "XII.", "XIII.", "XIV.", "XV.", "XVI.", "XVII.", "XVIII.", "XIX.", "XX.", "XXI.", "XXII.", "XXIII.", "XXIV.", "XXV.", "XXVI.", "XXVII.", "XXVIII.", "XXIX.", "XXX.", "XXXI.", "XXXII.", "XXXIII.", "XXXIV.", "XXXV.", "XXXVI.", "XXXVII.", "XXXVIII.", "XXXIX.", "XL.", "XLI.", "XLII.", "XLIII.", "XLIV.", "XLV.", "XLVI.", "XLVII.", "XLVIII.", "XLIX.", "L.", "LI.", "LII.", "LIII.", "LIV.", "LV.", "LVI.", "LVII.", "LVIII.", "LIX.", "LX.", "LXI.", "LXII.", "LXIII.", "LXIV.", "LXV.", "LXVI.", "LXVII.", "LXVIII.", "LXIX.", "LXX.", "LXXI.", "LXXII.", "LXXIII.", "LXXIV.", "LXXV.", "LXXVI.", "LXXVII.", "LXXVIII.", "LXXIX.", "LXXX.", "LXXXI.", "LXXXII.", "LXXXIII.", "LXXXIV.", "LXXXV.", "LXXXVI.", "LXXXVII.", "LXXXVIII.", "LXXXIX.", "XC.", "XCI.", "XCII.", "XCIII.", "XCIV.", "XCV.", "XCVI.", "XCVII.", "XCVIII.", "XCIX.")
alphabet_sections = ("A.", "B.", "C.", "D.", "E.", "F.", "G.", "H.")


# Define function to split outline into sections for content generation
def split_outline(outline):
    sections = {}
```

```python
    current_section = None

    # Split outline into lines
    lines = outline.split('\n')

    # Loop through lines
    for line in lines:
        # If line is a section header starting with Roman numerals, create a new section key
        if line.strip().startswith(roman_numerals):
            current_section = line.strip()
            sections[current_section] = []
        # If line is a subsection header starting with alphabets, add as a value to the last Roman numeral key
        elif line.strip().startswith(alphabet_sections):
            if current_section is not None:
                sections[current_section].append(line.strip())
        # If line is not a section header, append it to the current section value
        elif current_section is not None:
            # Check if the list exists before trying to append to it
            if sections[current_section]:
                sections[current_section][-1] += ' ' + line.strip()
            else:
                sections[current_section].append(line.strip())
    # Creates empty subsection for keys with no values
    for key in sections.keys():
        if not sections[key]:
            sections[key] = ['']

    return sections


######################################################
# Generate paragraphs
######################################################

```

```python
#Define function to generate text for each section given in the
    outline
def generate_paragraphs(sections):
    generated_text_list = []

    # Loop through each section header and subtopics
    for section_header, subtopics in sections.items():
        # Loop through each subtopic within a section
        for subtopic in subtopics:
            # Generate prompt
            prompt = f"Write a few paragraphs on the following
    thesis paper section: {topic}. {section_header}: {subtopic}."
            print(f"Writing section {section_header}: {subtopic}")
            # Generate text using OpenAI API
            responses = openai.Completion.create(
                model='text-davinci-003',
                prompt=prompt,
                temperature=0.8,
                max_tokens=3800,
                top_p=1,
                frequency_penalty=0,
                presence_penalty=0.6,
                best_of = 5
            )
            # Append the generated text to the list
            generated_text_list.append(f"{section_header}: {subtopic
    } \n\n {responses.choices[0].text} \n")

    # Join the generated text together to create the final string
    generated_text_draft =  '\n\n'.join(generated_text_list)
    # return list of generated text
    return generated_text_list, generated_text_draft



#####################################################
# Self assessment
```

```python
######################################################

# define function for scoring and providing feedback on generated
    text
def evaluate_sections(generated_text_list):
    evaluations = {}
    # Loop through each section
    for draft_section in generated_text_list:

        # Evaluate the response using OpenAI API
        prompt = f"Please evaluate the quality of the following
    section on a scale of 1-100, with 100 being the highest possible
    score. Evaluate based on the clarity of the writing, the
    coherence of the arguments, and the use of evidence to support
    claims;  list the score first, then provide an explanation for
    the score with specific improvements. Section:{draft_section} "
        print("Evaluating secton")
        response_eval = openai.Completion.create(
            model='text-davinci-003',
            prompt=prompt,
            temperature=1.2,
            max_tokens=500,
            top_p=0.7,
            frequency_penalty=0.5,
            presence_penalty=1.2,
        )

        # Append the generated text to evaluations dictionary
        evaluations[draft_section] = response_eval['choices'][0]['
    text']

    # Rename the dictionary storing sections and their feedback
    evaluations_dict = evaluations

    # Initialize string version of functionoutput
    evaluations_draft = ""

    # Add values to dictionary
    for key, value in evaluations.items():
         evaluations_draft += "\n\n" + key + "\n" + "\n SECTION FEEDBACK: " + value + "\n\n"

    return evaluations_dict, evaluations_draft


######################################################
# Score parsing
#####################################################

# define function to parse scores from feedback for later calculations
def get_scores(evaluations_dict):
    scores = []
    for feedback in evaluations_dict.values():
        score_str = ""
        for char in feedback:
            if char.isdigit():
                score_str += char
                if len(score_str) == 2:
                    if score_str == "10" and feedback[feedback.index(score_str) + 2] == "0":
                        score_str += "0"
                    break
        scores.append(score_str)

    return scores



# define functions to add scores to outline foe easy viewing for problem sections
def scored_outline(sections, scores):
    output_string = ""

    scores_with_indices = [(score, i) for i, score in enumerate(
   scores)]

    for section_num, (section_title, subsections) in enumerate(
   sections.items()):
        output_string += f"{section_title}\n"

        for subsec_num, subsection_title in enumerate(subsections):
            score, score_index = scores_with_indices[section_num*2 +
    subsec_num]
            output_string += f"{subsection_title}: {score}/100\n"

        output_string += "\n"  # add an empty line after each
   section

    return output_string



########################################################
# Save
########################################################
# Create a unique file name
def write_output_to_file(topic, outline, generated_text_draft,
   evaluations_draft, average_score, scored_outlines):
    # Create a unique file name
    timestamp = datetime.now().strftime("%Y-%m-%d-%H-%M-%S")
    filename = f"{topic}_paper_{timestamp}.txt"
    filepath = "./your/output_location/" + filename

    # Compile components to be saved as output
    full_text = f" Topic: {topic} \n\n Outline:\n\n  {outline} \n\n
   Draft: \n\n {generated_text_draft}. \n\n\n\n EVALUATIONS \n\n
   Scored Outline: {scored_outlines} \n\n OVERALL Score: {
   average_score} \n \n Draft with Feedback: {evaluations_draft}"

    # Save generated text to a new file with unique name
    with open(filepath, 'w', encoding='utf-8') as file:
        file.write(full_text)


#####################################################
# RUN FUNCTIONS
#####################################################

# Ask user for topic
topic = input("Please enter a topic: ")


# Make Outline
outline = generate_outline(topic)
print(outline)
print("Outline Complete.")



# Split into sections
sections = split_outline(outline)
print("Sectioning Complete.")


# Generate Paragraphs
generated_text_list, generated_text_draft  = generate_paragraphs(
    sections)
print("Generate paragraphs complete.")



# Evaluate
evaluations_dict, evaluations_draft = evaluate_sections(
    generated_text_list)

print ("evaluation complete")

```

```python

# Parse Scoring
scores = get_scores(evaluations_dict)
total_score = sum(int(score) for score in scores)
average_score = total_score / len(scores)
print("Average score=", average_score)
scored_outlines =  scored_outline(sections, scores)
print(scored_outlines)
print ("scoring complete")


# Write output to file
write_output_to_file(topic, outline, generated_text_draft,
    evaluations_draft, average_score, scored_outlines)




#######################################################
# end timing
#######################################################
end_time = time.time()
#print("Total time taken: {:.2f} seconds".format(end_time -
    start_time))
total_time = end_time - start_time

minutes, seconds = divmod(total_time, 60)
print("Total time taken: {:.0f} minutes {:.2f} seconds".format(
    minutes, seconds))
print("Please check output!")
```

Listing 1: Version model˙17.py

3.2 Outputs

Running the model generates a .txt file as output, stored in `./your/output_location/`.

3.2.1 Sample Topic: ”GPT-3”

NOTE: Model 17 outputs text in a .txt file. The output below is trimmed to only include sections with feedback to demonstrate capabilities of the model and specially formatted for this document. The complete version of this sample output along with others is available on the project GitHub repository.

BEGIN OUTPUT
“‘

```
Topic: GPT-3

Overall Score: 84.64285714285714

Scored Outline:

I. Introduction
    A. What is GPT-3?: 75/100
    B. Significance and Impact: 95/100

II. Overview of GPT-3
    A. Description: 85/100
    B. Architecture: 85/100
    C. Performance: 90/100

III. Applications of GPT-3
    A. Natural Language Processing: 90/100
    B. Machine Learning: 80/100
    C. Robotics: 90/100
    D. Autonomous Systems: 90/100

IV. Challenges of GPT-3
```

A. Privacy: 90/100
B. Security: 90/100
C. Reliability: 85/100

IV. Conclusion
A. Summary: 85/100
B. Future Directions: 85/100

Draft with Feedback:

I. Introduction: A. What is GPT-3?

GPT-3 is a natural language processing (NLP) model developed by OpenAI. It is the largest language model ever created and has been trained on a massive dataset of natural language. GPT-3 stands for Generative Pre-trained Transformer and it is the successor to GPT-2, a much smaller NLP model. GPT-3 uses a deep learning technique called transformers and is designed to generate human-like text from a given prompt. It is capable of producing natural language at a level that rivals professionals in certain tasks.

B. What are the Benefits of GPT-3?

GPT-3 provides many benefits over traditional NLP models. It can generate text that is more human-like and accurate than other models. The large size of GPT-3 allows it to capture long-range dependencies between words which can improve the accuracy of its predictions. Additionally, GPT-3 requires less data than other models, allowing it to be trained quickly and cost effectively. Finally, GPT-3's ability to generate text from a prompt makes it ideal for applications such as automatic summarization and question answering.

SECTION FEEDBACK:
Score: 75/100
Explanation: The section is generally well-written and the arguments are clear. How-

ever, it could benefit from more specific examples of how GPT-3 provides benefits over traditional NLP models and provide additional evidence to support these claims. Additionally, some words such as ”transformers” should be defined or linked to an external resource for readers who may not be familiar with the term. With these improvements, the score would increase to 100/100.

I. Introduction: B. Significance and Impact

GPT-3 (Generative Pre-trained Transformer 3), developed by OpenAI, is a revolutionary language model that has revolutionized the field of natural language processing. GPT-3 is a deep learning algorithm that uses a transformer-based architecture to generate human-like text. It is the largest language model ever created and has been trained on a dataset of over 45TB of text.

GPT-3 has been used in a wide range of applications, including question answering, natural language understanding, automatic summarization, and information retrieval. One of its primary benefits is its ability to generate human-like text without explicit programming. This has enabled developers to create powerful and sophisticated applications with minimal effort.

GPT-3's significance and impact have been felt across a range of industries. For example, it has been used for automatic customer service chatbots, conversational AI systems, personalized news curation, and natural language search. In addition, GPT-3 has been used to generate summaries of long documents and to perform automated translations from one language to another.

The success of GPT-3 has led to its widespread adoption by developers and companies. GPT-3 has been used by major technology companies such as Microsoft, Google, and Amazon to improve their products and services. Additionally, GPT-3 has become hugely popular among researchers and developers who are looking to create new applications with minimal effort.

Overall, GPT-3 is a powerful and revolutionary language model that has had a profound impact on the field of natural language processing. Its ability to generate human-like text without explicit programming has enabled developers to create powerful applications with minimal effort. Moreover, its widespread adoption by major technology companies and researchers has made it an essential tool for anyone looking to develop applications using natural language processing.

SECTION FEEDBACK:
Score: 95/100
This section is well-written and provides a good overview of GPT-3, its significance, and impact. The writer clearly explains the capabilities of GPT-3, how it has been used in various applications, and how it has been adopted by major technology companies. The arguments are well organized and supported with evidence. To improve the quality of this section, the writer could include more specific examples to further illustrate the impact of GPT-3 across different industries. Additionally, the writer could provide more details on how GPT-3 works internally to better explain its revolutionary nature.

II. Overview of GPT-3: A. Description

GPT-3 (Generative Pre-trained Transformer 3) is an AI machine learning model created by OpenAI and released in 2020. It is a natural language processing model that uses deep learning algorithms to learn from large amounts of text data. GPT-3 is trained on a dataset of 45TB of text, making it one of the largest language models ever created. GPT-3 is capable of producing humanlike text, making it an important tool for language understanding.

GPT-3 is based on a type of artificial neural network known as a Transformer. This type of network is used to process large amounts of data, such as text data, in an efficient way. GPT-3 is a generative model, meaning it can generate new text based on the input it is given. This makes it a powerful tool for natural language processing as it can generate text that is indistinguishable from human writing.

GPT-3 has a variety of applications including natural language understanding, question-answering, text summarization, and automated writing. It is also being used to create conversational agents and virtual assistants. GPT-3 has been praised for its incredible accuracy and the potential it holds for the advancement of artificial intelligence.

SECTION FEEDBACK:
Score: 85/100
Explanation:This section provides a good overview of GPT-3 and its applications. The writing is clear, the arguments are coherent, and evidence is used to support claims. To further improve this section, more detail could be provided on how GPT-3 works (i.e., its architecture) as well as additional examples of its applications. Additionally, more evidence could be included to back up the claim that GPT-3 has been praised for its accuracy and potential in AI advancement.

II. Overview of GPT-3: B. Architecture

GPT-3, or the Generative Pre-trained Transformer 3, is a neural network-based language model created by OpenAI. It is one of the most advanced natural language processing systems currently available, and it has been lauded for its impressive performance in many areas of artificial intelligence. GPT-3's architecture is based on an encoder-decoder model, which is used to generate text from language data.

The encoder-decoder model is composed of two components: the "encoder", which takes an input text and generates a representation of that text; and the "decoder", which then uses this representation to generate the desired output. GPT-3 contains over 175 billion parameters, including parameters for word embeddings and various layers of convolutional and recurrent neural networks. These parameters are used to create a powerful representation of language that can be used for a variety of tasks.

In addition to its encoder-decoder architecture, GPT-3 also contains a number of other features. For instance, GPT-3 includes a "Prompt Tuner", which allows the model to customize its input and output to better match the context of the task at hand. Additionally, GPT-3 also features "Automatic Tokenization", which can automatically convert words into tokens, allowing the model to more effectively process user queries.

Overall, GPT-3's architecture is very powerful, allowing the model to quickly process large amounts of data and accurately generate responses. This makes GPT-3 a powerful tool for natural language processing tasks, such as question answering, summarization, translation, and text generation.

SECTION FEEDBACK:
Score: 85/100
Explanation: The writing is clear and the argument is coherent. However, the use of evidence to support claims could be improved by providing more examples and details about how this information can help sports bettors make better decisions when placing their bets. Additionally, more detail could be provided about the calculation of expected frequency and probability for each outcome.

II. Overview of GPT-3: C. Performance

GPT-3 is a powerful language model developed by OpenAI, which stands for Generative Pre-trained Transformer. It is a transformer-based language model that uses deep learning to produce human-like text. GPT-3 has been trained on a massive dataset of text, making it able to generate highly accurate results when predicting the next word or phrase.

Performance is an important factor to consider when evaluating the effectiveness of GPT-3. In general, GPT-3 provides impressive performance in terms of accuracy and speed. On certain tasks such as natural language processing, GPT-3 can outperform humans in terms of accuracy. Additionally, GPT-3 can generate responses in near real-time, meaning it can provide fast results for users.

In terms of its ability to understand complex concepts, GPT-3 also performs well. In tests, GPT-3 was able to successfully complete tasks like text summarization and question-answering with only minimal input from the user. This indicates that GPT-3 is able to understand complex text, which is a promising sign for its potential applications.

Overall, GPT-3 provides impressive performance in both accuracy and speed. Its ability to understand complex concepts and generate responses in near real-time is a great advantage, making it a powerful tool for natural language processing tasks.

SECTION FEEDBACK:
Score: 90/100
Explanation: This section provides a clear overview of GPT-3's performance and explains its strengths in terms of accuracy, speed, and understanding complex concepts. The writer also provides evidence to support their claims by citing tests that demonstrate the model's capabilities. To further improve this section, the writer could provide more specific examples or studies that show GPT-3's impressive performance. Additionally, providing a comparison between GPT-3 and other models would help to further illustrate the effectiveness of GPT-3.

III. Applications of GPT-3: A. Natural Language Processing

GPT-3 (Generative Pre-trained Transformer 3) is a natural language processing (NLP) model developed by OpenAI. It is the most powerful and advanced NLP model to date, and it has been used in a variety of applications. In natural language processing, GPT-3 is used for tasks such as text generation, question answering, summarization, and many others.

GPT-3 has been used to improve the accuracy of existing NLP models. For example, GPT-3 has been used to improve the accuracy of sentiment analysis models, where it has been shown to outperform traditional methods. GPT-3 can also be used to generate textual responses from given questions or statements. This kind of capability has made it useful for virtual assistant applications and dialogue systems.

GPT-3 is also being used in more creative applications, such as writing poetry and generating short stories. GPT-3 has been trained on large datasets of literary works, and it can generate poetry and stories with similar themes and styles to classic authors. This has opened up new opportunities for authors and writers to explore different writing styles.

Finally, GPT-3 can also be used to generate content on its own. For example, GPT-3 has been used to generate articles on topics such as finance, healthcare, and sports. This has made it possible to quickly generate any kind of content without having to employ professional writers or editors.

SECTION FEEDBACK:
Score: 80/100
Explanation: This section is generally well-written and organized, with clear explanations of the applications of GPT-3. The arguments are coherent and supported by evidence. To improve the quality, there could be more detail on each application (such as specific examples) to make it easier for readers to understand how GPT-3 can be used in practice. Additionally, some technical terms could be defined or explained further to provide more clarity.

III. Applications of GPT-3: B. Machine Learning

GPT-3, or the Generative Pre-trained Transformer 3, is a language model developed by OpenAI. It is the largest language model ever created and can generate human-like text with a wide variety of applications. One of the most important applications of GPT-3 is in machine learning.

GPT-3 can be used to train machine learning models quickly and efficiently. It has been used to create text-based models that can generate paragraphs describing an image or predict the next word in a sentence. It can also be used for natural language processing tasks such as sentiment analysis, question answering, and text summarization.

GPT-3's ability to understand language makes it an ideal tool for training machine learning models. It can be used to generate large datasets with natural language inputs and outputs, which can then be used to train neural networks. This can drastically reduce the time and resources needed to develop powerful machine learning models.

GPT-3 has also been used in recent years to create intelligent chatbots and virtual assistants. These tools can be used to provide customer service, respond to queries, and help users complete tasks. In the near future, GPT-3 will likely play an even bigger role in how businesses interact with customers and how they build their automated systems.

SECTION FEEDBACK:
Score: 90/100
Explanation: This section provides a clear overview of the applications of GPT-3 in machine learning, supported by concrete examples and evidence. The arguments are well structured and coherent. The only potential improvement would be to provide more detailed information on the types of neural networks that can be trained with GPT-3, as well as any challenges or limitations associated with using it for machine learning tasks.

III. Applications of GPT-3: C. Robotics

Robotics is an area of research and development that has seen rapid growth in recent decades, and the emergence of GPT-3 has provided a powerful tool to enable further advances in this field. GPT-3 can be used to create robot models without the need for programming. This makes it possible to quickly create and test robot designs that are more complex than before. Additionally, GPT-3 can be used to build robots with natural language capabilities, enabling them to understand human speech and respond appropriately. Furthermore, GPT-3 can be used to create robots that are able to interact with their environment, helping them to navigate and complete tasks. Finally, GPT-3 can be used to improve the accuracy of robotic vision systems, allowing robots to better identify objects in their environment. Overall, GPT-3 provides a powerful tool to further advance robotics and create novel and innovative applications.

SECTION FEEDBACK:

Score: 90/100

Explanation: This section provides a clear and concise explanation of the applications of GPT-3 in robotics, including examples of how it can be used. The arguments are coherent and well-structured, and the use of evidence to support claims is adequate. However, there could be further detail provided on each application mentioned, such as providing specific examples or additional evidence to demonstrate its effectiveness. Additionally, more concrete examples of the potential benefits of using GPT-3 in robotics could also be included.

III. Applications of GPT-3: D. Autonomous Systems

Autonomous systems are becoming increasingly popular and GPT-3 is being used in many of these applications. For example, GPT-3 can be used to develop autonomous robots that can navigate by themselves. Autonomous robots powered by GPT-3 are capable of learning from their environment and adapting to their surroundings. GPT-3 can also be used to train autonomous vehicles like self-driving cars, which can drive themselves safely and efficiently.

In addition to robotics and autonomous vehicles, GPT-3 is also being used to create autonomous systems for natural language processing (NLP). NLP is used to understand and process human language, and GPT-3 can be used to create systems that can understand natural language and respond intelligently. This technology can be used to create chatbots and virtual assistants that can understand human language and respond appropriately.

Finally, GPT-3 can also be used to create autonomous systems for computer vision. This technology can be used to create systems that can identify objects in images and videos, as well as detect and track objects in real-time. This technology can be used to develop facial recognition systems, object recognition systems, and autonomous surveillance systems.

Overall, GPT-3 is being used to create a wide range of autonomous systems for robotics, autonomous vehicles, NLP, and computer vision. This technology has a wide range of potential applications and has the potential to revolutionize the way we interact with technology.

SECTION FEEDBACK:

Score: 85/100

Explanation: This section provides a comprehensive overview of the applications of GPT-3 and clearly explains how it can be used to create autonomous systems. The writing is clear and coherent, and evidence is provided to support the claims. However, there could be more detail provided on each application such as specific examples or use cases, which would help provide a better understanding of the potential applications of GPT-3. Additionally, more information could be provided on the implications of this technology for the future.

IV. Challenges of GPT-3: A. Privacy

GPT-3, which stands for Generative Pre-trained Transformer 3, is a powerful language model developed by OpenAI. It has the potential to revolutionize natural language processing and machine learning. However, GPT-3 also presents several challenges, one of which is privacy.

Privacy is a major concern with GPT-3 as it raises questions about data security and data ownership. As GPT-3 is trained on large datasets, there is a risk that user data could be exposed or used in unintended ways. Additionally, GPT-3 relies on cloud computing and other related services, which raises the risk of unauthorized access to the data. Moreover, the large amounts of data required to train GPT-3 can easily be abused by malicious actors, who can use the data for their own benefit.

To address these privacy issues, OpenAI has taken several steps. These include introducing access control mechanisms, as well as encrypting data and using secure authentication protocols. In addition, OpenAI has also set up an AI Safety Working Group to ensure that GPT-3 is transparent and secure. Despite these measures, there is still a need for more research and development to ensure that GPT-3 remains safe and secure.

In conclusion, privacy is one of the biggest challenges faced by GPT-3. OpenAI has taken measures to address this issue, but more needs to be done to ensure full data security. It is essential to ensure that user data remains protected and that malicious actors are unable to abuse the system.

SECTION FEEDBACK:
Score: 85/100
The section is clear and the arguments are coherent, however there could be more evidence to support the claims. Specific improvements include providing more examples of measures taken by OpenAI to address privacy issues, as well as further research or studies that have been conducted to improve data security for GPT-3. Additionally, a conclusion summarizing the main points of the argument could help to reinforce the overall message.

IV. Challenges of GPT-3: B. Security

The security of GPT-3 is one of its main challenges. As a powerful artificial intelligence system, GPT-3 has the potential to be used for malicious purposes. This could include generating automated accounts to spread false information, producing convincing images and audio to deceive people, or creating malicious code to hack into systems. The potential for misuse of GPT-3 is vast and could have serious consequences for society.

Furthermore, GPT-3 is an open source platform, meaning that anyone can access its algorithms and use them for their own purposes. This means that it is very difficult to control who is using the system and what they are using it for. GPT-3 could be used to manipulate public opinion, generate fake news, or create bots to carry out fraudulent activities.

It is also important to consider the potential for GPT-3 to be used to create "deepfakes", which are videos or images that have been altered to appear real. With GPT-3's ability to generate convincing images and audio, deepfakes could be used to misinform and deceive the public. This could have serious implications for our democracy as well as our personal safety.

Overall, the security of GPT-3 presents a significant challenge that must be addressed in order to ensure that it is being used responsibly. As GPT-3 is increasingly used in everyday life, strategies must be developed to protect against misuse and ensure that it is only being used for legitimate purposes.

SECTION FEEDBACK:

Score: 85/100

Explanation: This section provides a comprehensive overview of the security challenges associated with GPT-3, including examples of potential misuse and implications for democracy and safety. The writing is clear and well-structured, with arguments that are coherently presented. Evidence is used to support claims throughout the text. To improve this section, some specific strategies or solutions to address these challenges could be included to provide more concrete advice.

IV. Challenges of GPT-3: C. Reliability

The reliability of GPT-3 is one of its greatest challenges. GPT-3 is a powerful language processing tool that has the potential to revolutionize natural language processing and artificial intelligence, but it is not yet perfect. The reliability of GPT-3 is limited by its reliance on large datasets that can be biased or incomplete, which can lead to inaccurate results. Additionally, GPT-3 is vulnerable to malicious actors who can manipulate its data and use it for malicious purposes.

In order to increase the reliability of GPT-3, steps need to be taken to ensure the validity of its data sources. This includes vetting data sources for biases, inaccuracies, and other issues that could lead to unreliable results. Additionally, security measures need to be taken to ensure that GPT-3 can't be manipulated by malicious actors. Finally, GPT-3 needs to be tested against a variety of datasets in order to ensure its accuracy and reliability.

By taking these steps, GPT-3 can become a reliable and powerful tool for natural language processing and artificial intelligence. However, until these steps are taken, the reliability of GPT-3 will remain an issue. Therefore, it is important for developers and researchers to take steps to ensure that GPT-3 can be relied upon to produce accurate and reliable results.

SECTION FEEDBACK:

Score: 75/100

Explanation: The section is well-structured and provides clear arguments for the challenges of GPT-3 related to reliability. However, it could benefit from providing more concrete examples of how data sources can be vetted and what security measures need to be taken to protect against malicious actors. Additionally, further explanation on the types of datasets that should be used for testing would improve clarity.

V. Conclusion: A. Summary

The conclusion of this paper is that GPT-3 is a powerful and versatile tool for natural language processing. GPT-3 is a new type of language model that uses deep learning and statistical language modeling techniques to generate text. GPT-3 has been shown to be effective at generating text in a variety of domains, including natural language understanding, dialogue systems, and question answering. Additionally, GPT-3 is highly scalable and efficient, making it suitable for applications such as natural language generation, text summarization, and machine translation. Furthermore, GPT-3 can be used in a variety of settings, including in the enterprise, on mobile devices, and in the cloud.

Overall, GPT-3 is an important milestone in the field of natural language processing and is likely to have a significant impact on the future of language technology. GPT-3 is a powerful tool that can be used in many different applications, from natural language generation to text summarization and machine translation. Furthermore, GPT-3 is highly scalable and efficient, making it a great choice for both enterprise and mobile applications. All in all, GPT-3 has the potential to revolutionize the way we use language, making it easier for humans to communicate with machines.

SECTION FEEDBACK:

Score: 75/100

Explanation: The section provides a good overview of the capabilities of GPT-3 and is clearly written. The arguments are coherent, with the conclusion being effectively supported by evidence from earlier in the paper. However, there could be more detail given to support the claims made, such as examples of how GPT-3 has been used successfully in various applications or concrete benefits that it offers. Additionally, some transitions between paragraphs would help to improve the flow of the writing.

V. Conclusion: B. Future Directions

GPT-3 has opened a new door of possibilities when it comes to artificial intelligence development. While GPT-3 has already shown great potential in natural language processing, its capabilities are far from reaching their full potential. As technology advances, there are numerous future directions in which GPT-3 can be developed and improved.

One key future direction for GPT-3 is the development of personalized applications. GPT-3 has already been used to create personalized chat bots, but its capabilities can be further extended to other types of applications, such as virtual assistants, recommendation systems, and more. Additionally, GPT-3 can be used to create personalized learning models, allowing individual users to benefit from tailored learning experiences.

Another key future direction is the development of GPT-3-based decision-making systems. These systems could use GPT-3's natural language processing capabilities to help make informed decisions. For example, a GPT-3-based system could be used to help determine the best course of action in a given situation. This would allow organizations to make more informed decisions, resulting in increased efficiency and effectiveness.

Finally, GPT-3 can be further developed to enable the creation of more efficient natural language processing models. Currently, GPT-3 is limited in terms of its computational resources, but this could be improved with further research and development. Additionally, researchers are exploring ways to make GPT-3 models more efficient so that they can be used for more complex tasks.

In conclusion, GPT-3 has opened up a wide range of possibilities for artificial intelligence development. The potential for future development is immense, and GPT-3 is sure to continue playing an important role in the field of artificial intelligence for years to come.

SECTION FEEDBACK:
Score: 75/100
This section provides a clear overview of potential future directions for GPT-3, as well as the advantages they could provide. The arguments are coherent and supported with relevant evidence. To further improve this section, more specific examples of applications or decision-making systems could be provided to give readers a better understanding of the potential uses of GPT-3 in the future. Additionally, the conclusion could be strengthened by providing a clearer summary of how GPT-3 will continue to play an important role in AI development.

3.3 Model Limitations

This model does face some limitations in producing a perfect thesis. Some common issues in the otuputs were repetitiveness and lack of cohesion between the sections. The openai API enforces a 'token limit" every time the API is called. The token limit is why the thesis papers are generated section by section and is not able to generate an entire paper in just one call. Further, the feedback calls are only in-taking the section they re providing feedback on and not the entire paper.

Despite needing manual input to apply the provided feedback, this maintains a better quality of text for a thesis length paper (around 5,000 words here). Other model versions incorporated the feedback and re-wrote the paper for the user, however that commonly made many paragraphs over-specific and redundant.

References

[1] OpenAI. Openai api reference. `https://platform.openai.com/docs/api-reference/`, 2021. [Online; accessed 24-April-2023].

[2] Sida Wang, Percy Liang, Chris Manning, and Jiajun Wu. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2109.06114*, 2021.