Language Translation System using Neural Networks

MAJOR PROJECT - PPT
GROUP-157

GUIDE	Ms. Abhipsa Mahala
DETAILS	
NAME	**REGN NO.**
Saswat Seth	20010352
Ashish Kumar Das	20010418
SK Sangeet	20010270

C.V. Raman Global University

Outlines

Introduction 3-4

Identification and Description 5

Preparation Process 6-7

Literature Review 8

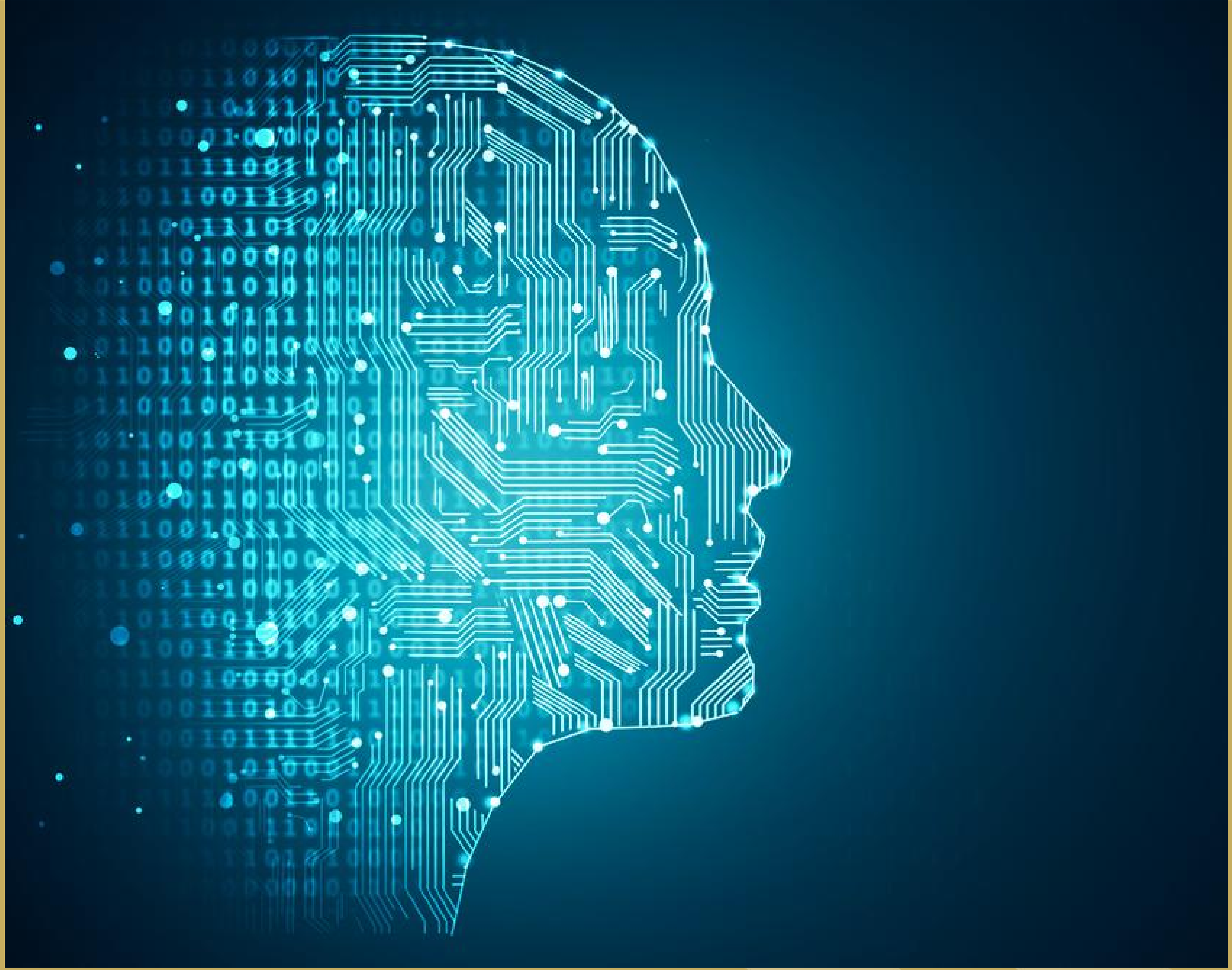
Summary of Previous Recommendations 9

Overview of Completed Tasks 10-15

Summary of Key Points 16

Conclusion 17

References 18



Introduction

Overview of Current Project Phase:

- We're midway through our project, gaining deep insights into various code components and complexities.
- Coding progress is notable, with crucial functionalities taking shape.
- Significant groundwork is done for core modules and features.
- Our team adeptly addressed initial challenges, strengthening the project's foundation.
- The ongoing phase focuses on refining and enhancing core functionalities with clear and elaborate code.

Core Modules:

- **Transformers Library:**
  - Purpose: Efficiently loads and manages the mT5-small model.
  - Functionality: Facilitates seamless integration of advanced translation capabilities.

A Language Translation System is a computer-based software or hardware solution that translates text or speech from one language to another. Created through the use of machine learning algorithms, neural networks, and large datasets.

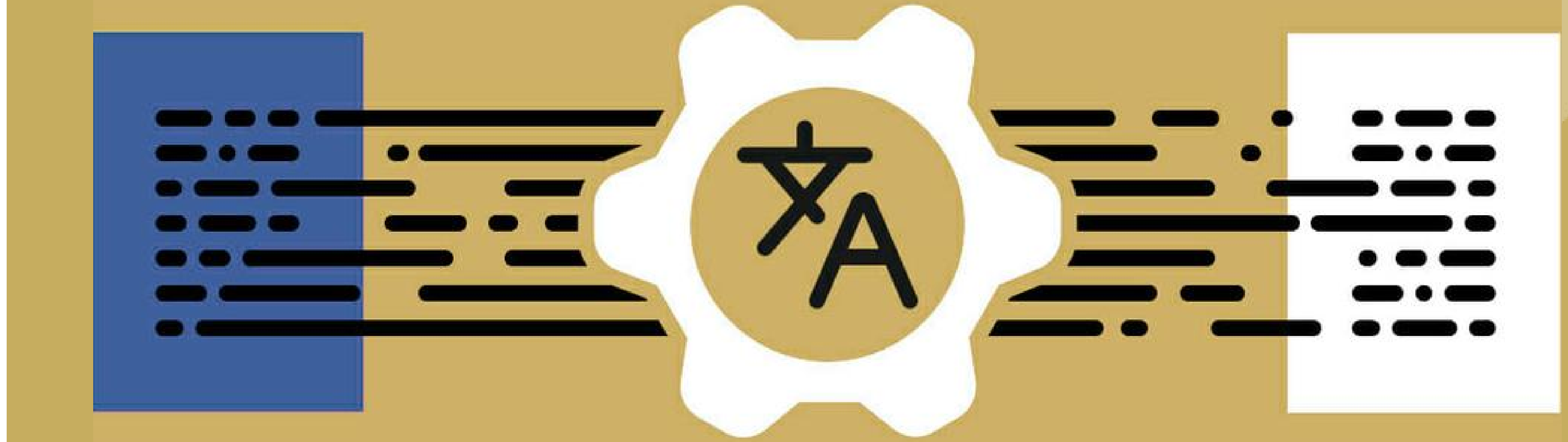
Introduction

- **SentencePiece Library:**
  - Role: Breaks down and tokenizes text.
  - Significance: Essential for effective processing of language inputs.
- **Datasets Library:**
  - Function: Provides diverse data for the translation model, enhanced by the alt dataset.
  - Importance: Ensures training on a wide range of language patterns.

Features and Functionalities:

- Multilingual Competence: Facilitates global communication through diverse language translation.
- Fine-tuning Precision: Ensures context-aware accuracy in domain-specific translations.
- Adaptability: Handles varied linguistic nuances, fostering a comprehensive understanding.
- Efficiency: Optimizes performance without compromising quality, thanks to mT5-small's compact design.
- Cross-cultural Communication: Adept at accommodating multiple languages for inclusive communication.
- Context Awareness: Exhibits advanced contextual understanding for improved accuracy.

Identification and Description



Test Bed Preparation/Platform Identification

- Model Selection: 'google/mt5-small' is chosen as the machine translation model.
- Tokenization: Text is tokenized and encoded for model input.
- Dataset Loading: The 'alt' dataset is loaded for training and testing.
- Data Splitting: Dataset is split into training and testing subsets.
- Language Mapping: Language tokens (e.g., 'en' to '<en>') aid language identification.
- Data Prep: Custom functions and data generators prepare data for training and testing.

Preparation Process

Selected Tools and Platforms:

- mT5-small Model: mT5-small enables translation across a diverse set of languages.
- Transformers & SentencePiece Libraries: Transformers and SentencePiece libraries enhance language input processing.
- Datasets Library with Alt Dataset: Datasets Library, along with Alt dataset, ensures thorough model training.
- Google Colab: promotes a collaborative workflow, fostering efficient teamwork.
- Hugging Face Repository: Hugging Face Repository provides pre-trained models and essential resources.

Level of Competence and Analysis

Expertise in mT5-small Model: In-depth understanding and effective utilization, showcasing proficiency in maximizing multilingual capabilities and leveraging advanced features.

Proficiency in Libraries: Competence in utilizing Transformers and SentencePiece libraries, highlighting efficiency in managing and processing language inputs.

Data Analysis Skills: Rigorous analysis of the Alt dataset within Datasets Library, showcasing the ability to discern patterns and ensure comprehensive model training.

Preparation Process

Collaborative Skills in Google Colab: Evident collaborative skills showcased through seamless teamwork and real-time collaboration in Google Colab.

Adaptability to Hugging Face Resources: Competence demonstrated by adapting and integrating resources from the Hugging Face Repository.

Compliance with Review-1 Recommendations:

Proactive Implementation: Demonstrating our commitment to continuous improvement, we've proactively refined and optimized the translation model for enhanced efficiency.

Future-Ready Adaptations: Anticipating areas of improvement, our forward-thinking approach ensures adaptability to future recommendations.

Self-Reflective Iterations: Through self-reflective iterations, we aim to elevate our project beyond the baseline, exceeding expectations for a future-proof language translation system.

Rigorous Testing: Our compliance strategy will involve rigorous testing and validation, ensuring our project's functionalities align with envisioned goals and provide a solid development foundation.

Literature Review

Name	Advantages	Disadvantages
• mT5: A massively multilingual pre-trained text-to-text transformer	• Multilingual capabilities. • Versatility in tasks. • Improved translation quality.	• May not excel in language-specific translation. • Model size may be a limitation. • Complex for specific tasks.
• BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	• Improved translation quality. • Leveraged pre-trained models. • Enhanced language understanding.	• Primarily designed for language understanding. • Adaptation may require substantial fine-tuning. • Not language-specific.

Summary of Previous Recommendations

Implementation and Integration

- Recommendations were diligently put into practice.
- System modifications were effectively integrated.
- CPU-centric processing became the core of our approach.
- A diverse range of languages was seamlessly incorporated.
- Cutting-edge model architectures were seamlessly integrated into the system.
- The groundwork for real-time translation is well underway.

Improvements in Sufficiency and Efficacy

- The implemented modifications have propelled the project to new heights.
- Translation precision and adaptability to new languages have significantly improved.
- These enhancements offer more accessible and accurate cross-lingual communication solutions.
- Our system has evolved to meet the increasing demands for efficient language translation.
- Improved efficiency and efficacy are now core attributes of our language translation platform.

Overview of completed tasks

Progress Since Last Review

Strong Foundation: Significant groundwork has been established for core modules, laying a robust foundation for further development.

Data Processing Configuration: Configured data processing steps, including tokenization, encoding, and dataset splitting. The Alt dataset was loaded, and language tokens were defined to map specific languages for translation.

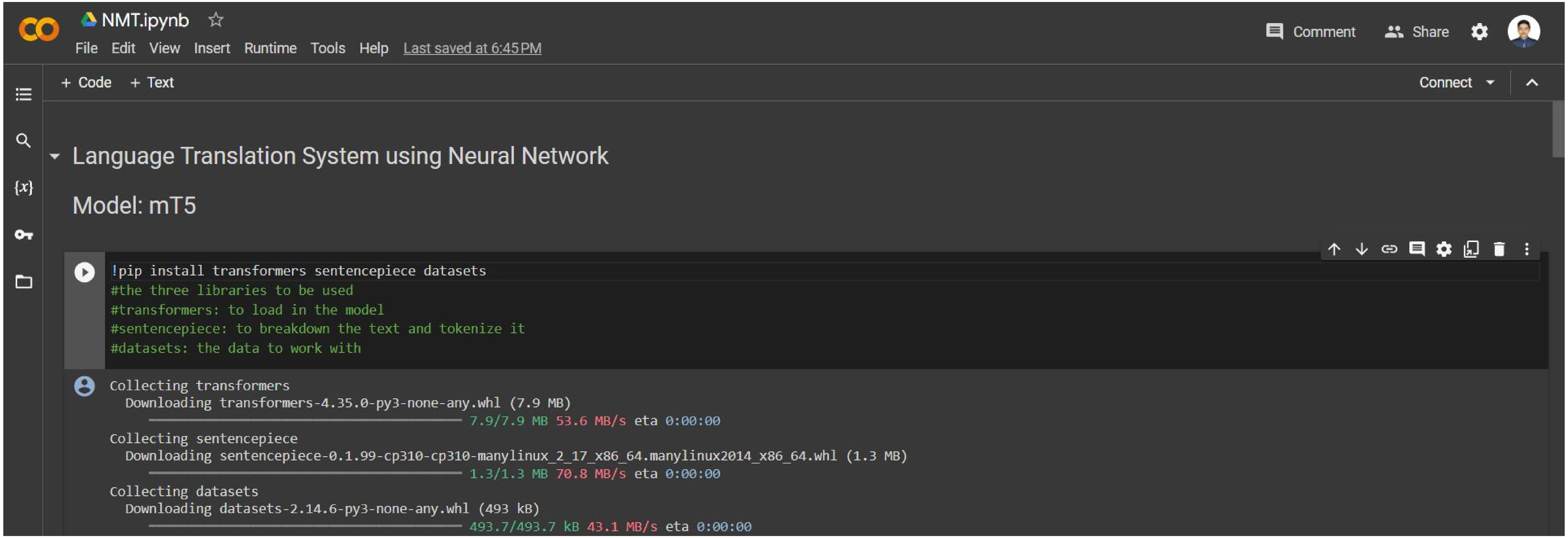
Refining Core Functionalities: Ongoing efforts focus on refining and enhancing core functionalities for a comprehensive and polished outcome.

Overcoming Challenges: The team has navigated initial challenges effectively, showcasing resilience and problem-solving skills.

Continuous Improvement: Proactive implementations and refinements demonstrate a commitment to enhancing overall efficiency and functionality.

Overview of completed tasks

Code Snippets



NMT.ipynb
File Edit View Insert Runtime Tools Help Last saved at 6:45 PM
Comment
Share
+ Code + Text
Connect

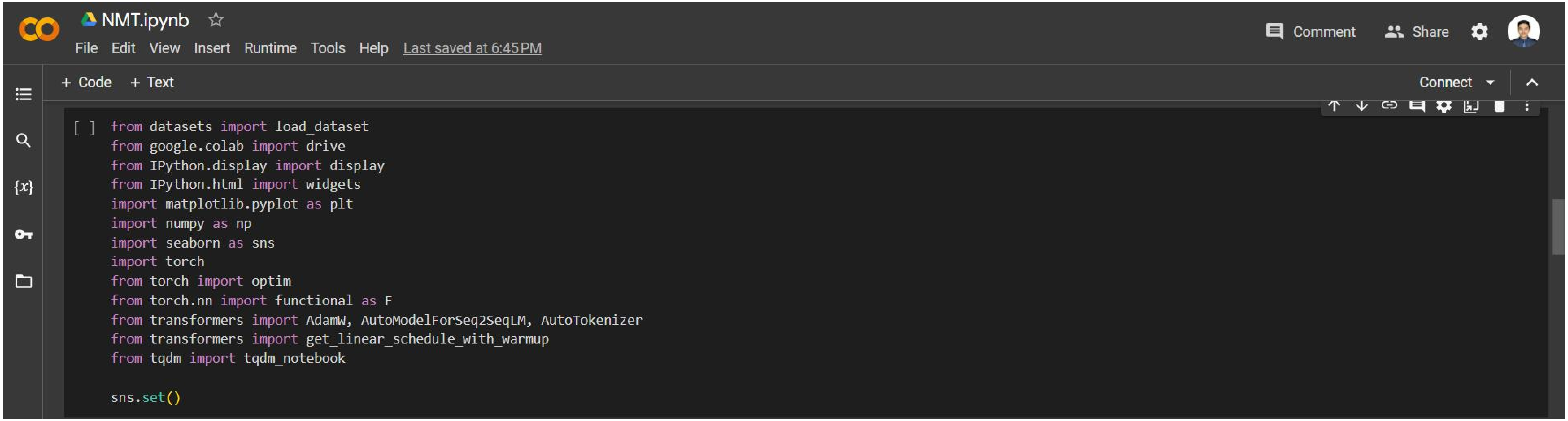
Language Translation System using Neural Network

Model: mT5

```
!pip install transformers sentencepiece datasets
#the three libraries to be used
#transformers: to load in the model
#sentencepiece: to breakdown the text and tokenize it
#datasets: the data to work with
```

```
Collecting transformers
  Downloading transformers-4.35.0-py3-none-any.whl (7.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.9/7.9 MB 53.6 MB/s eta 0:00:00
Collecting sentencepiece
  Downloading sentencepiece-0.1.99-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (1.3 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 70.8 MB/s eta 0:00:00
Collecting datasets
  Downloading datasets-2.14.6-py3-none-any.whl (493 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 493.7/493.7 kB 43.1 MB/s eta 0:00:00
```

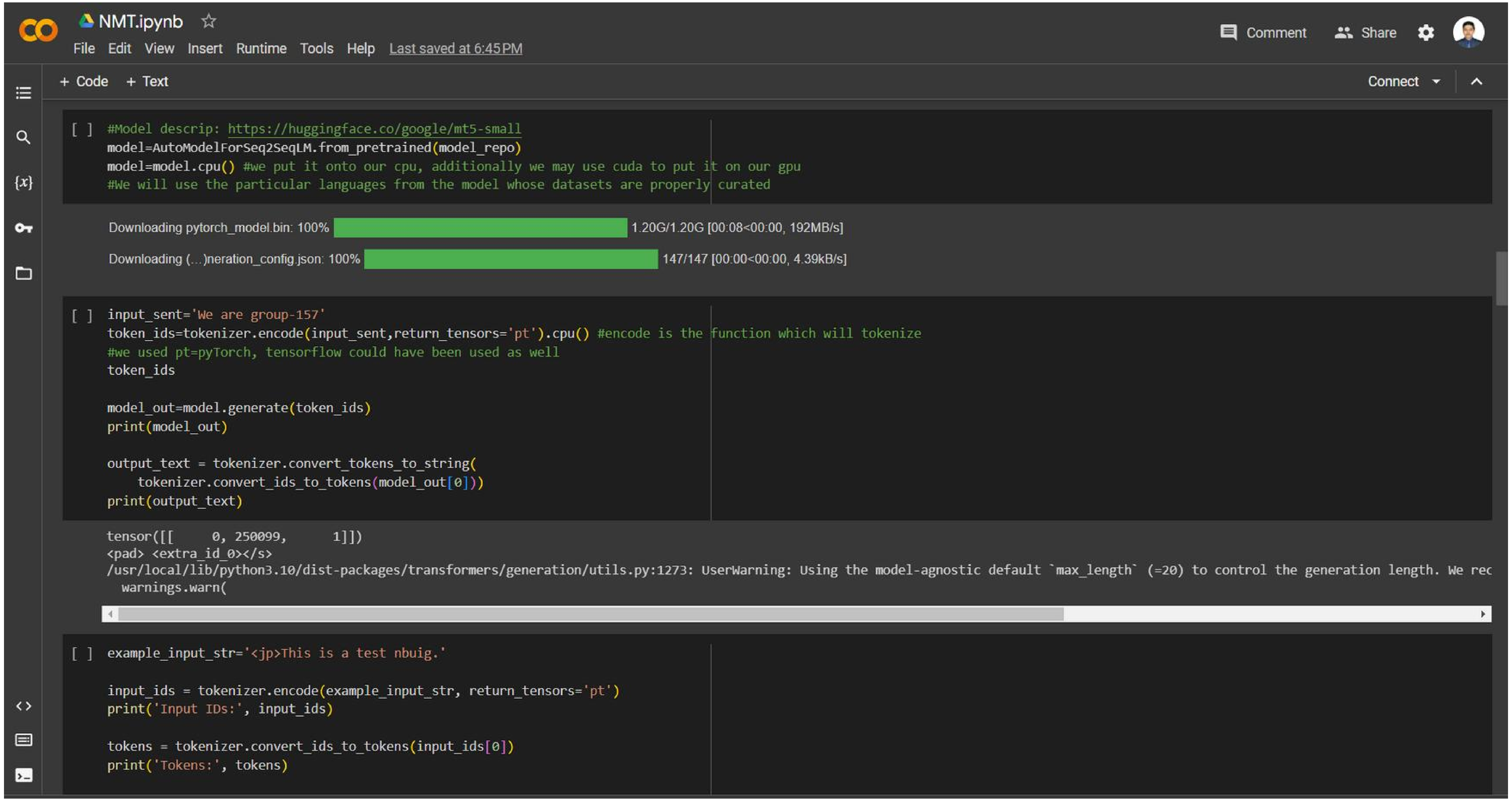
Overview of completed tasks


NMT.ipynb
File Edit View Insert Runtime Tools Help Last saved at 6:45 PM
Comment
Share
+ Code + Text
Connect
[ ]

```
from datasets import load_dataset
from google.colab import drive
from IPython.display import display
from IPython.html import widgets
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import torch
from torch import optim
from torch.nn import functional as F
from transformers import AdamW, AutoModelForSeq2SeqLM, AutoTokenizer
from transformers import get_linear_schedule_with_warmup
from tqdm import tqdm_notebook

sns.set()
```

Overview of completed tasks



```
[ ] #Model descrip: https://huggingface.co/google/mt5-small
    model=AutoModelForSeq2SeqLM.from_pretrained(model_repo)
    model=model.cpu() #we put it onto our cpu, additionally we may use cuda to put it on our gpu
    #We will use the particular languages from the model whose datasets are properly curated
```

```
Downloading pytorch_model.bin: 100% 1.20G/1.20G [00:08<00:00, 192MB/s]

Downloading (…)neration_config.json: 100% 147/147 [00:00<00:00, 4.39kB/s]
```

```
[ ] input_sent='We are group-157'
    token_ids=tokenizer.encode(input_sent,return_tensors='pt').cpu() #encode is the function which will tokenize
    #we used pt=pyTorch, tensorflow could have been used as well
    token_ids

    model_out=model.generate(token_ids)
    print(model_out)

    output_text = tokenizer.convert_tokens_to_string(
        tokenizer.convert_ids_to_tokens(model_out[0]))
    print(output_text)
```

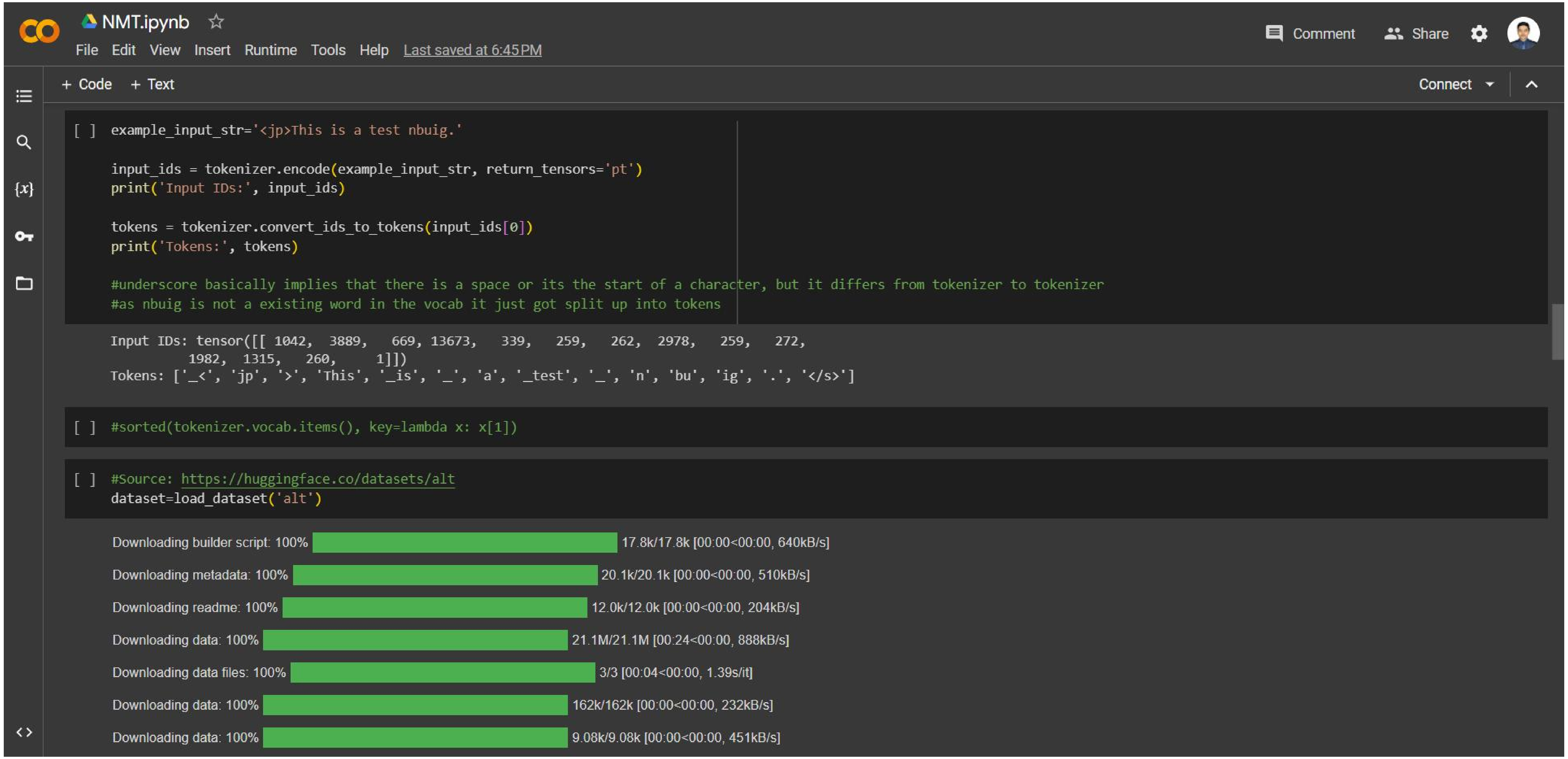
```
tensor([[     0, 250099,      1]])
<pad> <extra_id_0></s>
/usr/local/lib/python3.10/dist-packages/transformers/generation/utils.py:1273: UserWarning: Using the model-agnostic default `max_length` (=20) to control the generation length. We rec
  warnings.warn(
```

```
[ ] example_input_str='<jp>This is a test nbuig.'

    input_ids = tokenizer.encode(example_input_str, return_tensors='pt')
    print('Input IDs:', input_ids)

    tokens = tokenizer.convert_ids_to_tokens(input_ids[0])
    print('Tokens:', tokens)
```

Overview of completed tasks



```
[ ]  example_input_str='<jp>This is a test nbuig.'

     input_ids = tokenizer.encode(example_input_str, return_tensors='pt')
     print('Input IDs:', input_ids)

     tokens = tokenizer.convert_ids_to_tokens(input_ids[0])
     print('Tokens:', tokens)

     #underscore basically implies that there is a space or its the start of a character, but it differs from tokenizer to tokenizer
     #as nbuig is not a existing word in the vocab it just got split up into tokens
```

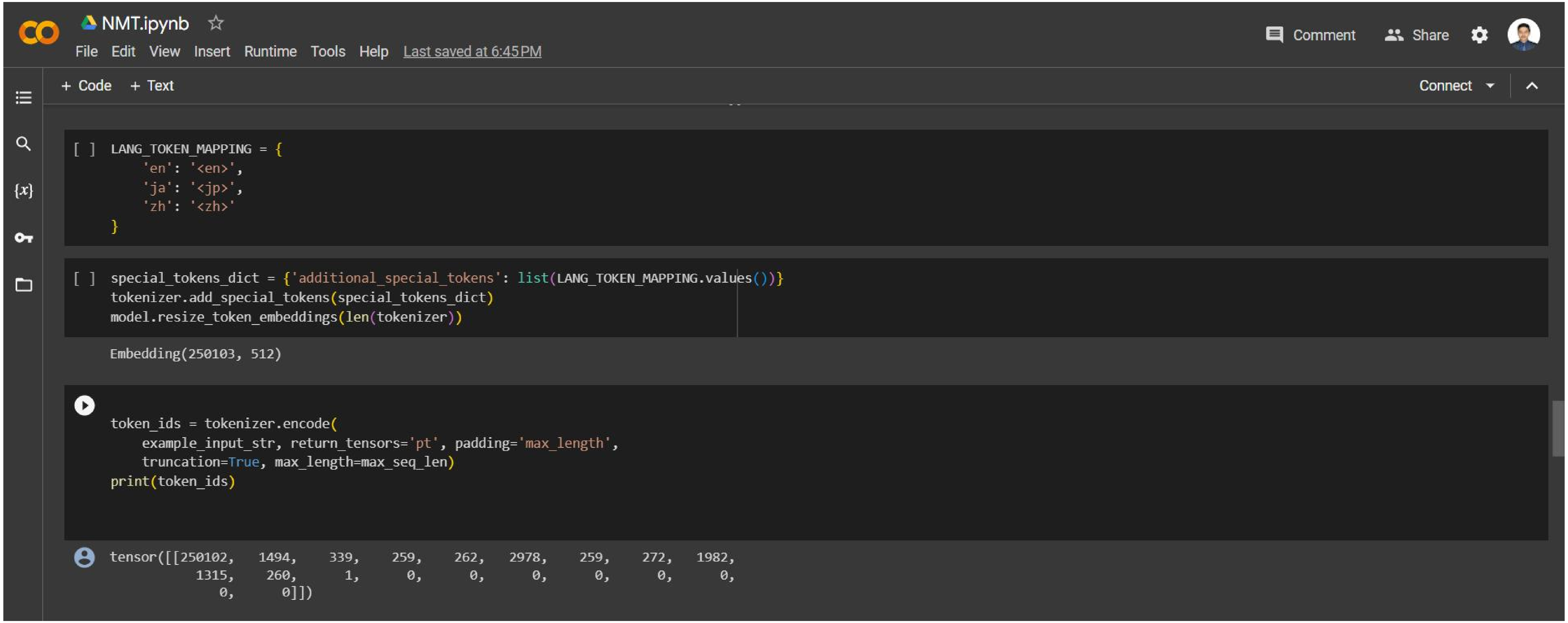
```
Input IDs: tensor([[ 1042,  3889,   669, 13673,   339,   259,   262,  2978,   259,   272,
          1982,  1315,   260,     1]])
Tokens: ['_<', 'jp', '>', 'This', '_is', '_', 'a', '_test', '_', 'n', 'bu', 'ig', '.', '</s>']
```

```
[ ]  #sorted(tokenizer.vocab.items(), key=lambda x: x[1])
```

```
[ ]  #Source: https://huggingface.co/datasets/alt
     dataset=load_dataset('alt')
```

```
Downloading builder script: 100%  17.8k/17.8k [00:00<00:00, 640kB/s]
Downloading metadata: 100%  20.1k/20.1k [00:00<00:00, 510kB/s]
Downloading readme: 100%  12.0k/12.0k [00:00<00:00, 204kB/s]
Downloading data: 100%  21.1M/21.1M [00:24<00:00, 888kB/s]
Downloading data files: 100%  3/3 [00:04<00:00, 1.39s/it]
Downloading data: 100%  162k/162k [00:00<00:00, 232kB/s]
Downloading data: 100%  9.08k/9.08k [00:00<00:00, 451kB/s]
```

Overview of completed tasks



NMT.ipynb
File Edit View Insert Runtime Tools Help Last saved at 6:45 PM
Comment Share
+ Code + Text
Connect

```
LANG_TOKEN_MAPPING = {
    'en': '<en>',
    'ja': '<jp>',
    'zh': '<zh>'
}
```

```
special_tokens_dict = {'additional_special_tokens': list(LANG_TOKEN_MAPPING.values())}
tokenizer.add_special_tokens(special_tokens_dict)
model.resize_token_embeddings(len(tokenizer))
```

```
Embedding(250103, 512)
```

```
token_ids = tokenizer.encode(
    example_input_str, return_tensors='pt', padding='max_length',
    truncation=True, max_length=max_seq_len)
print(token_ids)
```

```
tensor([[250102,   1494,    339,    259,    262,   2978,    259,    272,   1982,
           1315,    260,      1,      0,      0,      0,      0,      0,      0,
              0,      0]])
```

Summary

- Project Focus: Developing a Language Translation System using Neural Networks.
- Objectives: Enhancing precision and context awareness for specific language pairs.
- Innovation: Convergence of innovation and advancement in cross-lingual communication.
- Approach: Utilizing cutting-edge neural machine translation and sequence-to-sequence models.
- Language Support: Commitment to a wide array of languages and real-time translation capabilities.
- Applications: Enabling transformative applications in diverse industries.
- Impact: Working towards an interconnected and inclusive global society.

Conclusion

- The code has successfully prepared a test bed for machine translation.
- It selected an appropriate model, handled data, and implemented training.
- The model can be used for various translation tasks.
- Ongoing progress includes model fine-tuning and evaluation.
- Enhanced language translation technology.
- Multilingual versatility for diverse language pairs.
- Progress towards real-time cross-lingual communication.

Refernces

- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention Is All You Need. In Advances in Neural Information Processing Systems (pp. 30-40).

- Arivazhagan, N., Bapna, A., Firat, O., Lepikhin, D., Johnson, M., Krikun, M., ... Cherry, C. (2019). Massively multilingual neural machine translation in the wild: Findings and challenges. arXiv preprint arXiv:1907.05019.

- Liu, Z., Indra Winata, G., Madotto, A., & Fung, P. (2020). Exploring fine-tuning techniques for pre-trained cross-lingual models via continual learning. arXiv preprint arXiv:2004.14218.

- Carmo, D., Piau, M., Campiotti, I., Nogueira, R., & Lotufo, R. (2020). PTT5: Pretraining and validating the t5 model on Brazilian Portuguese data. arXiv preprint arXiv:2008.09144.

- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 4171-4186, Minneapolis, Minnesota. Association for Computational Linguistics.