International Journal of Education Humanities and Social Science

ISSN: 2582-0745
Vol. 6, No. 04; 2023

CTTMAP: A CORPUS TRANSLATION TEACHING MODEL ASSISTED BY PYTHON

Gan Xiaolan
School of Foreign Languages, Sichuan University of Science and Engineering, China

https://doi.org/10.54922/IJEHSS.2023.0556

ABSTRACT
This article presents the Corpus Translation Teaching Model Assisted by Python (CTTMAP), which integrates Python programming technology into corpus translation teaching. The model is based on constructivist learning theory, project-based teaching methods, and natural language processing. It aims to develop students' abilities to identify and analyze translation focal points and difficulties by utilizing Python technology. The model consists of 9 projects, categorized into basic syntax of Python, basic functions of NLTK, Python-based translation practice and rule projects. It incorporates teaching processes such as pre-class Python preparation, in-class Python analysis, and post-class Python exercises. The assessment heavily relies on the completion of these projects. The CTTMAP can be taught as an independent course or integrated into general or specialized translation teaching.

Keywords: CTTMAP; Python, Teaching model.

1. INTRODUCTION
The teaching model refers to a teaching method that is not only a teaching technique but also a comprehensive and systematic operational style. It covers aspects such as teaching principles, teaching objectives, teaching contents, teaching processes, and teaching assessments (Ye, 1993). This article will introduce a corpus translation teaching model assisted by Python (CTTMAP), which involves the integration of Python programming technology into corpus translation teaching. This teaching model can be taught as an independent course or selectively integrated into general or specialized translation teaching courses.

This teaching model is guided by constructivist learning theory, project-based teaching method, and natural language processing. It proposes to center the learner and utilize Python technology to cultivate students' abilities to identify and analyze translation focal points and difficulties. To achieve these objectives, the model incorporates 8 projects divided into Python-based translation practice projects and Python-based translation rule projects. Given that projects are the main content of this teaching model, the model designs teaching processes such as pre-class Python preparation, in-class Python analysis, and post-class Python exercises. Additionally, half of the assessment weight depends on students' completion of the projects. In all, CTTMAP can be described as figure 1 below.

2. TEACHING PRINCIPLES
2.1 Constructivism Learning Theory and Python
Constructivism was proposed by Swiss psychologist Jean Piaget. He conducted extensive research on children's cognitive development in the first half of the 20th century and proposed the constructivism learning theory. The main idea of this theory is that the world exists

ISSN: 2582-0745
Vol. 6, No. 04; 2023

objectively, but people's understanding and meaning attribution of the world are determined by individuals. (Piaget, 1950) Therefore, in the constructivist teaching process, teachers should respect and acknowledge students' existing knowledge and experience, and guide them to develop new knowledge and skills on this basis. This requires active interaction and cooperation between teachers and students. Teachers should facilitate interaction among students so that they can exchange ideas and experiences and learn new knowledge and skills from each other.

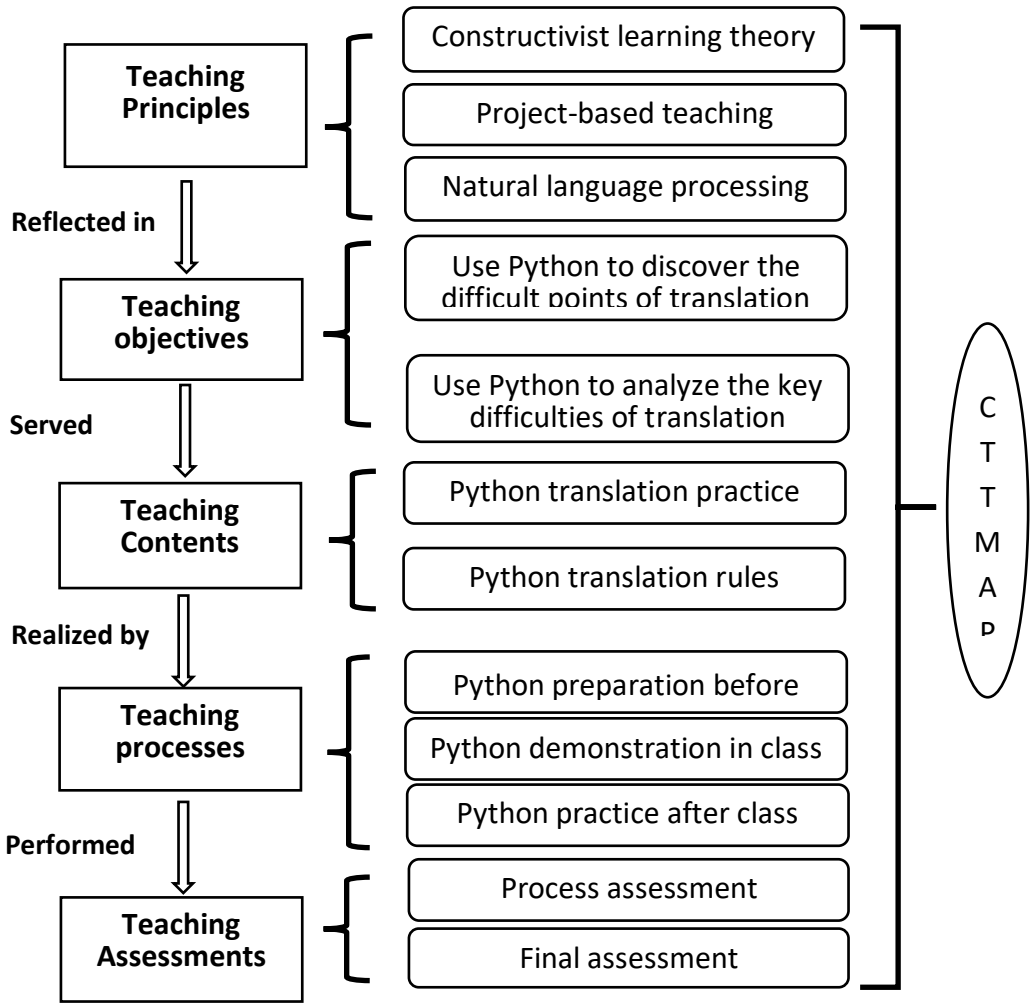

Figure 1 CTTMAP idea map

Python programming language has a close relationship with constructivist teaching. The learning process of Python emphasizes practice and exploration, and students construct knowledge by exploring problems and solving them through programming. This is very similar to the constructivist teaching philosophy, which emphasizes that students construct knowledge through their own practice and exploration. Python programming language also has the characteristics of openness and flexibility. Students can explore programming based on their own

ISSN: 2582-0745
Vol. 6, No. 04; 2023

interests and needs, thereby improving their initiative and participation in learning. At the same time, the Python community provides rich teaching resources and tools. Students can obtain help and exchange experiences through online learning platforms, programming competitions, community forums and other channels, thereby promoting interaction and cooperation among students. Therefore, Python programming language is very compatible with the constructivist teaching philosophy. Python programming can be regarded as a practical way of constructivist teaching, which has been widely used in the field of education.

2.2 Project-based teaching method and Python

The origin of project-based teaching method can be traced back to the educational ideas of John Dewey, an American education reformer. Dewey advocated for the integration of learning and practice, promoting student learning and development through practical experiences and problem-solving. He believed that students should construct knowledge and skills through hands-on activities and participation in social practices. (Dewey, 1938) The practice of project-based teaching has shown that students can better understand and apply their learning in projects, cultivate problem-solving abilities and innovative thinking, and enhance their motivation and interest in learning. Over time, project-based teaching method has gradually been applied and promoted worldwide. It has become an important teaching method in modern education, widely used in various disciplines and educational stages, providing students with more practical and comprehensive learning experiences.

Project-based teaching method also has a close relationship with Python because Python is a programming language that is highly suitable for project-based teaching. In Python project-based teaching, students can choose topics of their interest, such as developing simple games, writing data analysis scripts, building web applications, etc., and learn the fundamentals and advanced features of Python language through practical project practice. It also helps foster innovative thinking, teamwork, and practical skills. Therefore, project-based teaching method and Python have a close relationship, as they can mutually reinforce each other, providing students with more valuable learning experiences and development opportunities.

2.3 Natural Language Processing and Python

Natural Language Processing (NLP) primarily focuses on language computation, language resource construction, machine translation and machine-assisted translation, handwriting and printed text recognition, speech recognition and text-to-speech conversion, information retrieval, text classification, multimedia retrieval with natural language as the pivot, data mining, machine learning, knowledge acquisition, knowledge engineering, artificial intelligence research related to language computation, linguistic research related to language computation, social computing, and humanities computing.

Python Natural Language Processing primarily utilizes the Natural Language Toolkit (NLTK) developed by the University of Pennsylvania. NLTK was created as a research and teaching tool for NLP, and it provides a collection of extensive public datasets and models with easy-to-use interfaces, including functions such as tokenization, part-of-speech tagging, named entity recognition, and syntactic parsing.

ISSN: 2582-0745
Vol. 6, No. 04; 2023

2.4 Summary

Constructionist learning theory, project-based teaching method, and natural language processing all emphasize learner-centered learning approaches and are closely connected to Python. Therefore, using Python-assisted corpus translation teaching requires the application of these theories as principles and starting points for instruction. In CCTMAP, students should be the main participants in translation teaching. Due to individual differences among students, in this teaching, teachers can use Python technology to help students discover learning materials of varying difficulty levels in the corpus based on their actual abilities. This allows students to independently observe and analyze the application of translation strategies and explore specific translation patterns.

3. TEACHING OBJECTIVES

3.1 Understanding of Python's Basic Syntax

Understanding Python's basic syntax is a fundamental objective in CTTMAP. This objective is crucial because it lays the foundation for students to effectively use Python for language translation tasks. By achieving a strong grasp of Python's basic syntax, students will be equipped with the necessary skills to effectively utilize Python in their corpus translation endeavors. This understanding will enable them to write efficient and concise code, leverage Python's extensive library ecosystem, and explore advanced techniques in natural language processing. Ultimately, students will be well-prepared to tackle the challenges and complexities of corpus translation with the aid of Python.

3.2 Understanding of NLTK’s Basic Functions

Understanding the basic functions of the Natural Language Toolkit (NLTK) is another key objective in the Python-assisted corpus translation teaching model. NLTK is a powerful library for natural language processing (NLP) tasks, and it provides a wide range of functionalities that can greatly assist in language translation. Through hands-on exercises and projects, students will apply NLTK's functionalities to real-world translation scenarios. They will learn how to utilize NLTK to process language data, extract relevant information, and gain insights into translation challenges. By gaining a solid understanding of NLTK's basic functions, students will be equipped with the necessary tools to effectively leverage NLTK in their corpus translation projects. Overall, understanding the basic functions of NLTK empowers students to utilize its powerful NLP capabilities in their translation endeavors. NLTK's functionalities enable students to perform various linguistic analyses, extract relevant information, and make informed decisions during the translation process. This understanding enhances their ability to handle complex language data and produce high-quality translations.

3.3 Utilizing Python to identify the key points and challenges in translation

Utilizing Python in the process of translation can greatly assist in identifying the key points and challenges in translation. Utilizing Python in the process of translation can greatly assist in identifying key points and challenges in translation. Python's string manipulation capabilities, natural language processing functionalities, terminology management, and machine learning capabilities can be used to analyze the text, compare translations, manage terminology, and

ISSN: 2582-0745
Vol. 6, No. 04; 2023

automate the translation process, providing valuable insights into the translation process and helping to produce high-quality translations.

3.4 Utilizing Python to analyze the reasonable conclusions in translation

Analyzing and interpreting language corpora is a crucial step in many language-related tasks, from translation to language teaching. Python can greatly aid in this process, offering a variety of tools and libraries for text analysis and natural language processing. Python provides a variety of tools and libraries that aid in the analysis, interpretation, and reporting of language corpora. By leveraging these tools, students can perform text cleaning, statistical analysis, data visualization, and machine learning tasks, all of which contribute to a deeper understanding of the language data. Through this process, students can draw reasonable conclusions from the data and make informed decisions based on their insights, enhancing their subject matter expertise and language-related skills.

3.5 Summary

In line with these instructional objectives, this teaching model adopts a project-based teaching approach, breaking down the objectives into specific translation projects. For example, the projects could include getting acquainted with Python and NLTK, understanding Python's basic syntax, exploring the application of Python in language translation practices, and discovering translation patterns using Python, among others.

4. TEACHING CONTENTS

According to the principle of starting from easy to difficult, the teaching modules of CTTMAP are divided into four modules, covering nine projects. Specific projects can be found in Table 1.

Table 1 Modules and projects of CCTMAP

Projects	Teaching Contents	Teaching hours
Part I Python's Basic Syntax		
Project 1	1 Python's Basic Syntax 1.1 Variables, data types, operators, control flow statements 1.2 Functions and modules	2
Part II NLTK's Basic Functions		
Project 2	2.1 Tokenization, part-of-speech tagging, stemming and lemmatization 2.2 NER, text classification, sentiment analysis and language modeling	2
Part III Basic application of Python in translation teaching		
Project 3	3 Application of Python in vocabulary translation teaching 3.1 Comparison and translation of word meanings in English and Chinese 3.2 Choice of word meaning in translation	4
Project 4	4 Application of Python in sentence translation teaching 4.1 Translation of long sentences 4.2 Translation of difficult sentences	4
Project 5	5 Application of Python in discourse translation teaching	4

ISSN: 2582-0745
Vol. 6, No. 04; 2023

	5.1 Use of Passage Connectives 5.2 Coherence of Passages	
Part IV Advanced applications of Python in translation teaching		
Project 6	6 Application of Python in readability analysis of the texts 6.1 Readability analysis of English texts 6.2 Analysis of readability of Chinese texts	4
Project 7	7 Application of Python in sentiment analysis of the texts 7.1 Sentiment analysis of English texts 7.2 Sentiment analysis of Chinese texts	4
Project 8	8 Application of Python in comparative analysis of the texts 8.1 Comparison of similarities of text 8.2 Term Equivalence of Text	4
Project 9	9 Application of Python in quantitative analysis of translation style 9.1 Quantitative analysis of translation style 9.2 Quantitative analysis of the translator's style	4
Total		32

Note: The first and second part is a general knowledge that applies to all translation courses. The third part is suitable for general courses such as *Translation Theory and Practice*, *English-Chinese Translation*, and *Chinese-English Translation*; The last part is suitable for special courses such as *Literary Translation*, *Scientific and Technical Translation*, *Legal Translation*, and *Business Translation*.

4.1 Grasping Python's Basic Syntax

To achieve the first objective mentioned, students will start by familiarizing themselves with variables, data types, operators, control flow statements (such as if-else statements and loops).

Once students have grasped the basics, they will delve deeper into more advanced concepts, such as lists, dictionaries, and tuples, which are essential data structures in Python. They will learn how to manipulate these data structures to store and retrieve information efficiently. Additionally, students will explore file handling in Python, enabling them to read and write data from and to external files, which is particularly useful when working with language corpora.

Moreover, students will learn about functions and how to define their own functions in Python. They will understand the importance of modular programming and how to break down complex tasks into smaller, reusable functions. Students will also explore the concept of libraries and modules, such as the Natural Language Toolkit (NLTK), which provides a wide range of tools and resources for natural language processing and text analysis.

4.2 Masting NLTK's Basic Functions

To achieve the second objective, students will begin by exploring the core functionalities of NLTK. They will learn how to install NLTK and download the necessary datasets and resources. Students will then dive into various NLP tasks that NLTK supports, such as tokenization, part-of-speech tagging, stemming, lemmatization, and named entity recognition.

ISSN: 2582-0745
Vol. 6, No. 04; 2023

Tokenization is the process of breaking text into individual words or tokens. Students will learn how to use NLTK to tokenize sentences and paragraphs, which is essential for subsequent analysis and processing. They will also understand the importance of tokenization in translation tasks, as it allows for more accurate analysis and manipulation of language data.

Part-of-speech tagging involves assigning grammatical tags to words in a sentence, such as noun, verb, adjective, etc. Students will explore NLTK's part-of-speech tagging capabilities and understand how it can be used to analyze the syntactic structure of sentences. This information can be used to identify potential translation challenges, such as ambiguous words or phrases.

Stemming and lemmatization are techniques used to reduce words to their base or root forms. Students will learn how NLTK can be used to perform stemming and lemmatization, which can be helpful in normalizing language data and reducing the complexity of translation tasks. They will also understand the limitations and considerations when applying these techniques in different language contexts.

Named entity recognition (NER) is the process of identifying and classifying named entities, such as names of people, organizations, locations, etc., in text. Students will explore NLTK's NER capabilities and understand how it can be used to extract important information from language data. This can be particularly useful in translation tasks that involve translating proper nouns or domain-specific terminology.

In addition to these core functionalities, students will also be introduced to NLTK's capabilities for text classification, sentiment analysis, and language modeling. These advanced functionalities can provide valuable insights and assist in translation tasks that require deeper analysis and understanding of language data.

4.3 Identifying the Key Points and Challenges in Corpus Translation

There are several ways Python can contribute to the third objectives.

Firstly, Python's string manipulation capabilities can be used to identify key points in a source text. For instance, students can write Python code to analyze the frequently occurring words or phrases, which can highlight the main themes or arguments of the text. They can also identify word patterns, collocations, and expressions that convey essential meanings. By applying these techniques to the source text, students can gain insights into the key points and nuances that need to be conveyed in the translation.

Secondly, Python can be used to evaluate and compare the source and translated texts. By comparing the similarity and dissimilarity between the two texts, students can identify translation challenges and areas where improvement is needed. In this case, Python's natural language processing functionalities, such as word vectors, semantic similarity, and language models, can be used to compare the texts quantitatively. Students can write Python code to calculate metrics, such as precision, recall, and F1 score, which can help them objectively evaluate the quality of the translation. The metrics can also provide feedback to improve the translation process and identify common translation errors.

Thirdly, Python can be used in the process of terminology management. One of the main challenges in translation is managing terminology consistently and accurately. Python can be used to create and manage a terminology database, which can be used to identify key terms and

ISSN: 2582-0745
Vol. 6, No. 04; 2023

ensure their consistent use throughout the translation process. Students can write Python code to extract key terms from the source text, map them to the target language, and track their use in the translation. They can use NLTK's functionalities to identify and extract terms consistently, and create regular expressions to match synonyms and variants.

Finally, Python can be used to perform automatic translation and post-editing, which can greatly assist in identifying key points and challenges in translation. Students can utilize Python's deep learning capabilities to build machine learning models that can learn from existing translations and provide suggestions for improving the translation process. They can also apply post-editing tools, such as translation memory and machine translation engines, to improve efficiency and consistency in the translation process. Python can also be used to evaluate the quality of these tools and improve their performance over time.

4.4 Analyzing the reasonable Conclusions from Corpora

Firstly, text cleaning is an important step in preparing language corpora for analysis. This involves removing unnecessary characters and stop words, as well as stemming or lemmatizing words to reduce the number of unique words in the corpus. Python's NLTK library provides pre-built functions for these tasks. By cleaning the corpus, students can analyze the text more efficiently and accurately.

Secondly, analyzing language data involves applying various statistical techniques to extract relevant information. Python provides numerous libraries for statistical analysis, including NumPy and pandas. These libraries enable students to perform various statistical analyses, such as frequency counts, descriptive statistics, and inferential statistics. Additionally, Python's visualization libraries, such as Matplotlib and Seaborn, can be used to create informative charts and graphs that help to convey the data in a clear and concise way.

Thirdly, interpreting the results of analysis requires subject matter expertise and domain knowledge. Students must be able to draw reasonable conclusions from the data, identify trends and patterns, and make informed decisions based on their insights. Python can assist with this process by automating some of the data interpretation tasks. For example, Python's machine learning libraries, such as Scikit-learn and TensorFlow, can be used to build models that predict certain phenomena based on the data. Students can utilize these models to test hypotheses and develop insights.

Fourthly, students can use Python to generate reports that present their findings in a clear and concise manner. By utilizing Python's document generation libraries, such as Pandoc and LaTeX, students can create professional-looking documents that summarize their analysis and present their conclusions. They can also automate the report generation process, enabling them to produce reports quickly and easily.

4.5 Summary

Through the above modules and projects, the Python Corpus Translation Teaching model provides a systematic learning path, starting from the basics and skills, moving on to analysis and evaluation, and finally to automation and post-editing. By studying these modules and

ISSN: 2582-0745
Vol. 6, No. 04; 2023

projects, students can grasp the core concepts and techniques of corpus translation using Python, thereby improving translation quality and efficiency.

5. TEACHING PROCESSES

Python can be integrated into three phases of translation teaching: pre-class preparation, in-class demonstration, and post-class practice. In the pre-class preparation phase, students can use free Python compilers to familiarize themselves with Python application modules and related content in advance. In the in-class demonstration phase, a teacher-led, student-centered, and interactive teaching mode can be adopted. Firstly, the teacher presents representative teaching materials using Python, guiding students to observe and establish cognitive associations between form and meaning. Then, through concrete examples, students are organized into groups to discuss and complete relevant projects, followed by classroom presentations. Finally, the teacher demonstrates how to successfully complete translation projects using Python. In the post-class practice phase, the teacher assigns translation exercises that require students to utilize Python software to improve the quality and efficiency of their translations. The specific teaching process is illustrated in Figure 2.

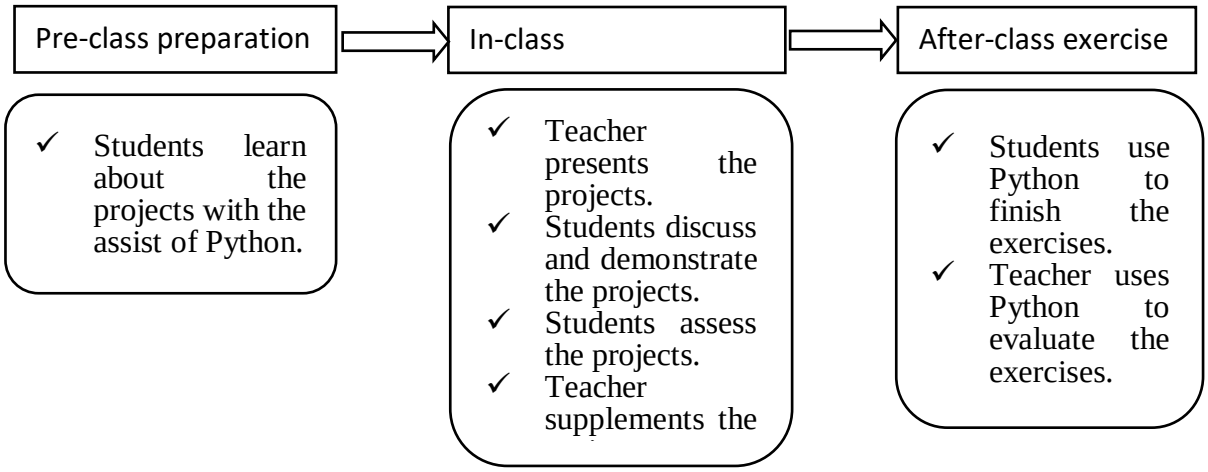

Figure 2: Teaching process of CTTMAP

By incorporating Python into pre-class preparation, in-class demonstration, and post-class practice, students can enhance their understanding and application of Python in translation, ultimately improving their translation skills and efficiency.

6. TEACHING ASSESSMENT

When teaching a dedicated course, the grading will be based on a percentage scale. A minimum of 60% is considered a passing grade. The overall course grade is composed of two parts: regular assessments and a final exam. The specific scoring and details are as follows:

Regular Assessments: 45% of the total grade, consisting of 9 projects, each worth 5%.

ISSN: 2582-0745
Vol. 6, No. 04; 2023

Final Exam: 55% of the total grade, either completing one exam paper or translating a horizontal project in the business sector.

7. CONCLUSION

By applying Python programming technology to corpus translation teaching, more practical and efficient tools and methods can be provided to help students better utilize corpus resources in the translation process. In CTTMAP, students can learn how to use Python programming software for text processing, data analysis, and automated translation skills. Teachers can guide students to understand the concept of corpus, the techniques of building and managing corpora, and how to use Python programming technology to process and analyze corpus data through practical operations.

ACKNOWLEDGMENTS

This paper is one of the achievements of the project (22CJRH10606) granted by Translators Association of Sichuan.

REFERENCES

1. Ye, Lan. (1993). New Pedagogy Course. Shanghai: East China Normal University Press.
2. Piaget, Jean. (1950). The psychology of intelligence. Routledge & Kegan Paul.
3. Dewey, John. (1938). Experience and education. Kappa Delta Pi.