

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA



2019-20

B.TECH, CSE

INDUSTRIAL TRAINING REPORT

MACHINE TRANSLATION EVALUATION

SUBMITTED BY:-

Chitransh Mishra

(17103103)

MENTOR:-

Dr. K. Vimal Kumar

CSE Department, JIIT

ACKNOWLEDGEMENT

I would like to thank my mentor **Dr. K. Vimal Kumar** , Department of CSE, Jaypee Institute of Information Technology, Sector-62, Noida. I would like to express my sincere gratitude towards him, as his guidance, encouragement, suggestions and constructive criticism have contributed immensely to the evolution of ideas on the project.

Turning this idea into a project wouldn't have been possible if he hadn't provided me with the knowledge he possesses and help me to get the best conclusion possible.

Chitrang Mishra

17103103

STUDENT DECLARATION

I hereby declare that the presented report under the **Industrial Training Programme** by Jaypee Institute Of Information Technology, Noida, titled “**Machine Translation Evaluation**” is an authentic record of work carried out under carried out under the supervision of **Dr. K. Vimal Kumar**. I have not submitted this work elsewhere for any other degree or diploma. I am fully responsible to the contents of this report.

Chitrang Mishra

Bachelor of Technology

Department of Computer Science,

Jaypee Institute of Information Technology

SUPERVISOR CERTIFICATE

This is to certify that **Chitrang Mishra (17103103)**, student of Bachelor of Technology, Computer Science, **Jaypee Institute of Information Technology, Noida** worked under my supervision during the Industrial Training Programme. I am pleased to state that he worked hard preparing the project and concerned reports. The information and findings in the report are authentic up to my knowledge.

He possess a good moral character and pleasing personality individually. I wish them every success in life.

Dr. K. Vimal Kumar

Assistant Professor

Department of Computer Science

Jaypee Institute of Information Technology, Noida

CONTENT LIST

1. Abstract	6
2. Introduction	7
WER	8
3. Related Work	9
4. Proposed Approach	11
Designing an evaluation system	11
Generating test cases	13
Calculating the efficiency score	14
5. Results	16
Search space	17
Efficiency Charts	18
6. Conclusion	19
7. Future Work	20
8. References	21

1. ABSTRACT

The language, which we use as a medium to communicate and exchange information of all forms, has forms which vary from one place to another. Even in the very same language, one can observe difference in accents, writing styles and general sense of perception. Computational linguistics is an interdisciplinary field concerned with the statistical or rule-based modeling of natural language from a computational perspective, as well as the study of appropriate computational approaches to linguistic questions. In common terms, it is the study of rules that govern the structure of human languages and model computer programs to understand and apply into various applications.

One of the many applications is machine language translation, which centers around the concept of translating sentences from one language to another without losing the general sense. This has been the principle behind services like Google Translate, Bing Translate and many more, focused at providing easy-to-go translation facilities to its users. Comparing the results of these services and scoring them accordingly is one big part of machine translation evaluation.

These services use their model, trained with millions of sentences and their appropriate translations, to provide user with appropriate translation results. We are evaluating these services by comparing their output with dictionary translation for various languages and effectively calculating the score for various services and calculating their efficiency in calculating the outputs. In summary, the proposed framework is flexible and generalizable, allows for efficient learning and scoring, and provides an MT evaluation metric that correlates with human judgments, and is on par with the state of the art.

2. INTRODUCTION

The evaluation of machine translation (MT) systems is a vital field of research, both for determining the effectiveness of existing MT systems and for optimizing the performance of MT systems. Traditionally, there are two paradigms of machine translation evaluation:

- (1) Glass Box evaluation, which measures the quality of a system based upon internal system properties, and
- (2) Black Box evaluation, which measures the quality of a system based solely upon its output, without respect to the internal mechanisms of the translation system.

Glass Box evaluation focuses upon an examination of the system's linguistic coverage and the theories used to handle those linguistic phenomena. Individual linguistic components of the system may be examined and be subjected to black box evaluations. This method of evaluation was primarily focused on rule-based expert systems, rather than statistical systems.

Black Box evaluation, on the other hand, is concerned only with the objective behavior of the system upon a predetermined evaluation set. This method of evaluation is only a fair comparison of systems if the systems being tested were both designed to work on data that is of the same character as the evaluation set or, if not, the person testing the systems has the objective of testing robustness across different data types with variations in structure, genre, and style.

Automatic machine translation evaluation metrics were developed due to the high costs, lack of repeatability, subjectivity, and slowness of evaluating machine translation output using human judgments, and the desire to enable automatic tuning of system parameters. The quality of these evaluation metrics is usually measured by

determining the correlation of the scores assigned by the evaluation metrics to scores assigned by a human evaluation metric, most commonly fluency and adequacy.

Word Error Rate (WER):

Word Error Rate (WER) is the standard metric of Automatic Speech Recognition performance and one of the first automatic metrics applied to machine translation. WER is computed as the Levenshtein distance (Levenshtein 1966) between the words of the system output and the words of the reference translation divided by the length of the reference translation.

The Levenshtein distance is computed using dynamic programming to find the optimal alignment between the MT output and the reference translation, with each word in the MT output aligning to either 1 or 0 words in the reference translation, and vice versa. Those cases where a reference word is aligned to nothing are labeled as deletions, whereas the alignment of a word from the MT output to nothing is an insertion. If a reference word matches the MT output word it is aligned to, this is marked as a match, and otherwise is a substitution. The WER is then the sums of the number of substitutions (S), insertions (I), and deletions (D) divided by the number of words in the reference translation (N) as shown in equation below:

$$WER = \frac{S+I+D}{N}$$

WER score is the most significant parameter towards calculating the efficiency of machine translation algorithms.

3. RELATED WORK

Machine translation is a prominent sub-field of research under Natural Language Processing (NLP). NLP has been a major field of contribution for scholars across to find a more efficient algorithm for computers to identify human vocabulary. NLP involves speech as well as text translation methods using various algorithm to define a more reasonable output.

1. “Research on the Relations Between Machine Translation and Human Translation” is a renowned research by Zhaorong Zong of School of Foreign Languages, Huangshan University, China.

2. Amazon’s Seattle Research Centre, home for Natural Language Processing for their premium assistant service, Alexa, is also a major centre for research on MT.

3. IIT Bombay and IIIT Hyderabad host a seperate research centre for understanding Indian languages and for research on MT. Jadavpur University’s Natural Language Processing Laboratory is also a major player.

4. Machine translation has been a sport for big players like Google and Microsoft in developing support for real-time translation services like Google Translate, Bing Translate and many others. These applications are used by small, medium and large enterprises. Some organizations offer multi-domain translation services—that is, customizable solutions across multiple domains—and other organizations offer translation solutions only for a specific domain. These solutions, although automated for the most part, still depend on human translators for pre- and post-editing processes.

5. These machine learning applications perform instant translation for textual, audio, and image files (images of words on screens, papers, signboards, etc.) from a source language into a target language. These are typically generic—that is, non-domain specific—albeit with a high translation accuracy.

6. Dublin-based KantanMT offers “a SaaS-based Machine Translation platform that enables users to develop and manage custom machine translation engines in the cloud.” According the company’s website, the platform can be customized to offer machine translation services across eight domains, including e-retail, government, travel, etc.

7. SYSTRAN offers a variety of translation services for five domains/industries. The SYSTRAN Enterprise Server, which is deployed on the business server of the client, offers three modes of translation:

- Full-text translation (translates plain text),
- File translation (translates files like Powerpoint, Excel, Word, etc., without altering their formats), and
- Web translation services (translates entire web pages).

4. PROPOSED APPROACH

Machine translation evaluation is aimed at finding the efficiency of algorithm of various machine translation systems and evaluating the systems on various other parameters. The most useful algorithm on comparison is the WER discussed earlier. WER requires two strings for comparison: one from the machine translation system, another which is a human translation of the subject string. The idea is to compare the outputs of various online MT service system with human translation. Accordingly, the project is divided into three stages:

- Designing an evaluation system.
- Generating test cases.
- Calculating the efficiency score.

4.1. Designing an Evaluation System:

This segment of the project focuses at designing an evaluation system to find the efficiency of a MT system. DeepL Translate, Google Translate, Bing Translate and Yandex Translate are the major services which this project lay focus on. Out of which, DeepL translate is rated to be the most close to dictionary level translation. The algorithm used by DeepL translate is rated by TechCrunch is rated to be the most efficient when it comes to using dictionary translation, hence, DeepL translation is established as a comparison parameter in the evaluation system.

A set of common English sentences is fed into the database which is translated by DeepL translate to various languages and stored in the database as a comparison set for other services. This task is performed by a python server hosted on heroku systems. The program uses selenium library to scrape data from various online system's website

using the inbuilt support of chromium webdriver. It also uses BeautifulSoup4 for separating important data from the complete html parse tree. Selenium and BeautifulSoup4 converts the obtained html page into a parse tree like the diagram shown below.

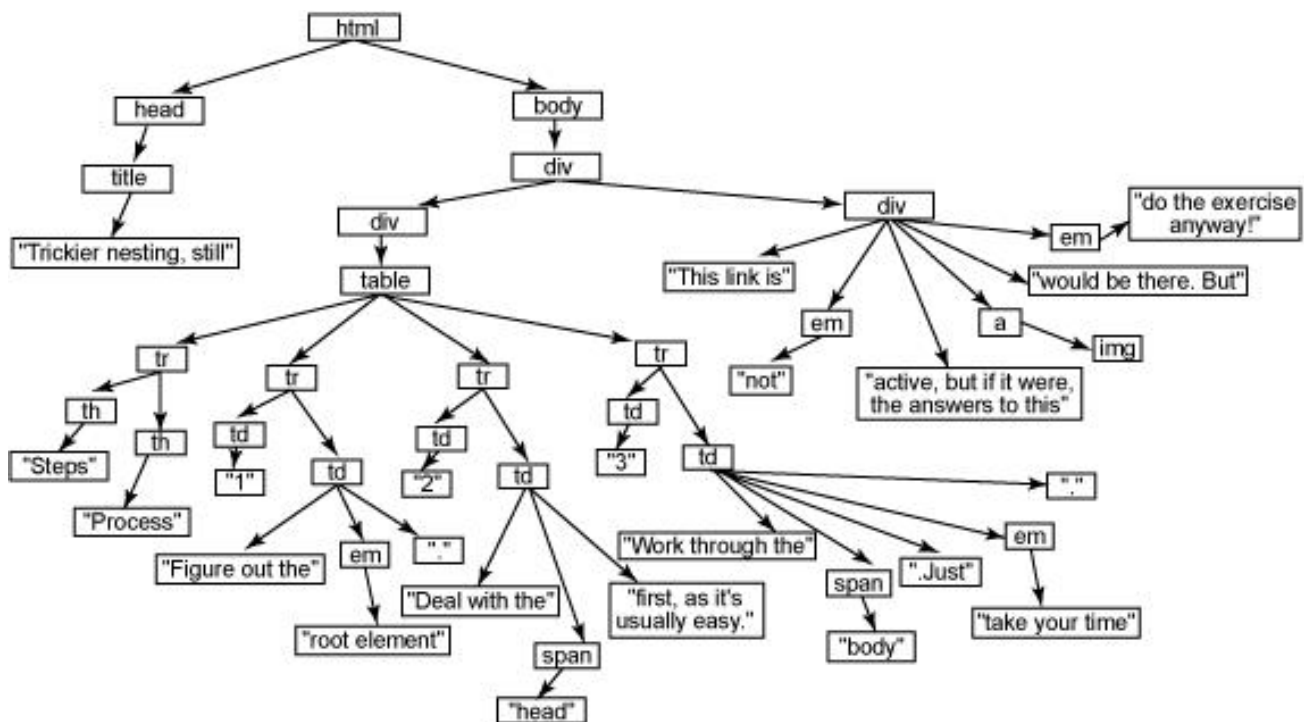


Fig. 4.1.1. HTML Parse Tree

The python code perform multiple requests on the DeepL Translate website with links containing the source and target language codes and the translation phrase. The HTML page is fetched, parsed and traversed for the required node. The text obtained is stored into the database for further processing. This stored set of translated statements act as the comparison set.

The database is hosted on mongoDB Atlas server and connected to the heroku server using pymongo library. Flask framework is used to design the python server. It is designed to iterate the steps required to calculate the parameters at a fixed time of the day.

4.2. Generating test cases:

The set of statements used in the making the comparison set are now translated using other MT systems i.e. Google, Bing and Yandex and the output is compared with the corresponding translation stored in the database. This is main training model which perform requests for each phrase in the database with each pair of source and target languages on each MT system.

For example:

Phrase: An apple a day, keeps a doctor away.

Source language:		Target language:
English	==>	German, Spanish, Italian, French
German	==>	English, French, Spanish, Italian
French	==>	English, German, Spanish, Italian
Italian	==>	English, French, Spanish, German
Spanish	==>	English, French, German, Italian

Number of MT systems: {Google, Bing, Yandex} = 3

Number of translation request performed: $4*5*3 = 60$ Translations per phrase.

These many translation request limits the speed of the server and opens door to a lot of error and exception raises. Therefore, server is designed such that it resumes processing from the point it left the translations. A practical record of all processes is stored in the database so as not to lose any result in between. Sampling a database of 500 sentences will require 30000 translation requests. If each request takes 2.5 seconds to process properly, each iteration of computing the efficiency will require 20+ hours to result (variables being network speed and server processing).

4.3. Calculating the efficiency score:

The Damerau–Levenshtein distance is a string metric for measuring the edit distance between two sequences. Informally, the Damerau–Levenshtein distance between two words is the minimum number of operations (consisting of insertions, deletions or substitutions of a single character, or transposition of two adjacent characters) required to change one word into the other. This algorithm is used to compare the two inputs i.e. MT system’s translated text and DeepL’s dictionary translation output.

The algorithm in python is defined below:

```
def compareString(s1, s2): # edit distance algorithm
    dp = [[0 for i in range(len(s1)+1)] for j in range(len(s2)+1)]
    for i in range(0, len(s1)+1):
        dp[0][i] = i
    for i in range(0, len(s2)+1):
        dp[i][0] = i

    for i in range(1, len(s2)+1):
        for j in range(1, len(s1)+1):
            if(s1[j-1] == s2[i-1]):
                dp[i][j] = dp[i-1][j-1]
            else:
                dp[i][j] = min(dp[i-1][j], dp[i-1][j-1], dp[i][j-1])+1
    return dp[len(s2)][len(s1)]
```

Passing the strings into the above functions returns the number of actions (insertion, replacement or deletion) required to convert one string into another. This number is used as a score for the trainer model.

For example : **compareStrings**(“levenshtein” , “meilenstein”)

This makes the following table in the program, called the dynamic programming top-down table.

		m	e	i	l	e	n	s	t	e	i	n
l e v e n s h t e i n	0	1	2	3	4	5	6	7	8	9	10	11
	1	1	2	3	3	4	5	6	7	8	9	10
	2	2	1	2	3	3	4	5	6	7	8	9
	3	3	2	2	3	4	4	5	6	7	8	9
	4	4	3	3	3	3	4	5	6	6	7	8
	5	5	4	4	4	4	3	4	5	6	7	7
	6	6	5	5	5	5	4	3	4	5	6	7
	7	7	6	6	6	6	5	4	4	5	6	7
	8	8	7	7	7	7	6	5	4	5	6	7
	9	9	8	8	8	7	7	6	5	4	5	6
	10	10	9	8	9	8	8	7	6	5	4	5
	11	11	10	9	9	9	8	8	7	6	5	4

Fig. 4.3.1. DP Top-Down Table

The last cell in the table represents the total number of actions required to be performed, i.e. the score for comparison.

This value is inserted into the below formula to calculate the efficiency of the MT system:

$$\text{Efficiency}(X) = \frac{\text{Efficiency}(X-1) * (X-1) + (L - S) / L}{X}$$

X - Iteration in action

L - Length of the original text

S - Score after comparison.

The above formula accumulates the efficiency into an efficiency matrix for each pair of source and target languages. After all iterations, the data is pushed onto the MongoDB server, which is displayed on the website in form of bar graphs.

5. RESULTS

The final results are available at

<https://chitrunk0614.github.io/MLE-Dialect-Analysis/>

The complete website source files are available at

<https://github.com/chitrunk0614/MLE-Dialect-Analysis>

The website is made using HTML, CSS and Javascript. It uses Bootstrap4 as a base framework for styling and Axios for making a promise-based request system. The data made available using API calls to the heroku server and MongoDB Atlas database.

The interface of the website is shown below:



Fig. 5.1. Project Homepage

A link to the Github repository is available on the homepage. The buttons provides opens spaces that perform their specific task:

1. Search Space

2. Efficiency Charts


5.1. Search Space


This opens a space which provides an option to search your query on various servers. The provided input will be fetched for translation from selected source to target language using various MT systems and the results will be displayed into the space provided, along with the time required for each request to complete and source link. The results are available for comparison as shown below:


Using the below options you can translate a single phrase from one language to another using various translators like [DeepL Translate](#), [Google Translate](#), [Bing Translate](#), [Yandex Translate](#). We will soon be increasing support for many other available translation services. The callback time is the time taken by the request to return a response and fetch a genuine result from the document received.

Phrase to Translate:

From: To:

 Bing Translate:
Scraped from: [Bing Translate Source](#) Response Time: 0.6728885173797607 seconds

 DeepL Translate:
Scraped from: [DeepL Translate Source](#) Response Time: 0.6844568252563477 seconds

 Google Translate:
Scraped from: [Google Translate Source](#) Response Time: 0.005448102951049805 seconds


 Yandex Translate:
Scraped from: [Yandex Translate Source](#) Response Time: 0.3987457752227783 seconds

Fig. 5.1.1. Search Space

All languages supported by the system are available in the dropdown. These translations use the API deployed on heroku. Official APIs are not used to avoid inconsistency in calculating the request time. All the results are scraped from the official websites.

5.2. Efficiency Charts

This option opens a space which displays charts for comparison of various services on two basis:

1. Comparing various MT Systems for ability to translate from a specific language to other target languages.

From English

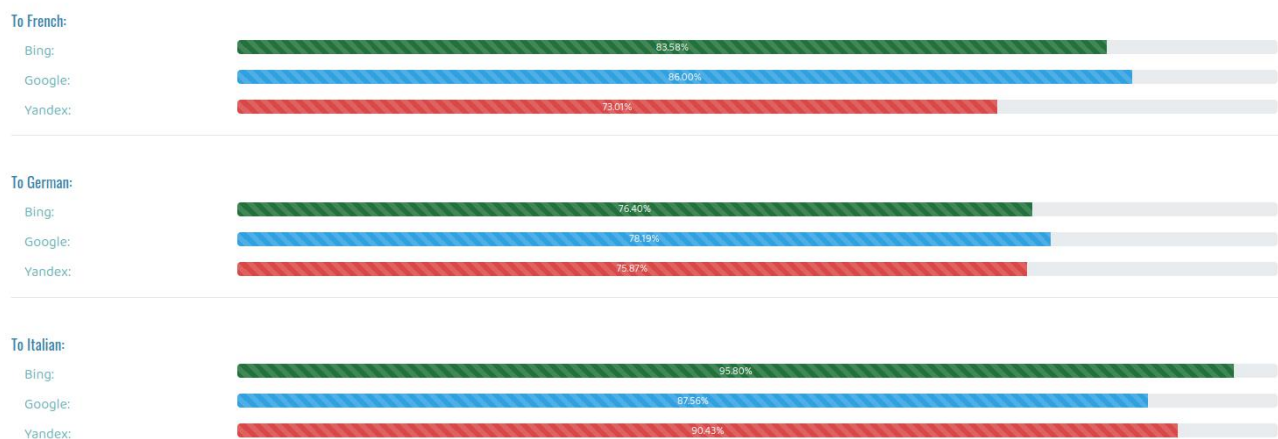


Fig. 5.2.1. Efficiency Charts (Translator Comparison)

2. Comparing various pair of languages for a specific MT system.

Bing Translate:

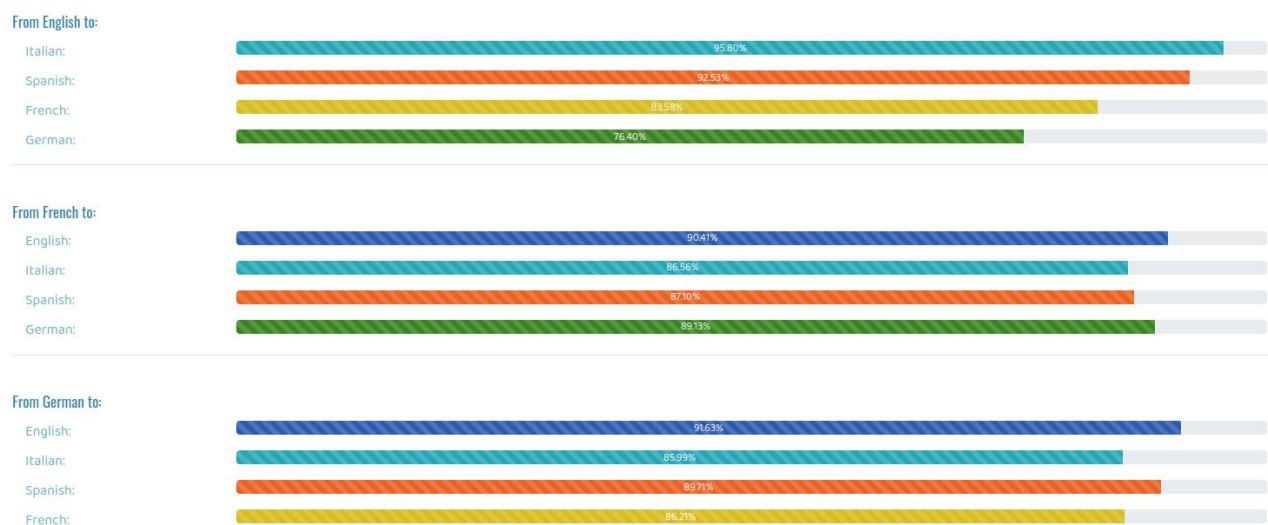


Fig. 5.2.2. Efficiency Charts (Language Comparison)

6. CONCLUSION

This project is completely independent working system. It has achieved all the objectives which were targeted:

- It can scrape all the MT systems for obtaining the outputs.
- It can compare the obtained results and calculate the efficiency.
- It provides an option to run a manual search for comparison through all the translation systems.
- It provides results as a visual data and helps ease the comparison.
- It is independent of the changes in the translation system's models. Efficiency modifies according to the changes of updates in the translator's model.

Through this project, I have achieved the following skills:

- I have learned the concepts of Machine translation, it's principles, it's applications and purpose and contribution towards modern technology.
- I have learned to use Selenium with python for scraping and unit testing the software.
- I have learned how a browser parse HTML script and how JavaScript promise model works.
- I have learned to make my own server, design a backend support system for a website, made my own APIs and deployed them on a global channel.
- I have learned to set timeline for objectives and work to achieve them within time.
- I learned to modify standard algorithm for the purpose of making an efficient backend service.
- I have learned to designing a complete user-end product and releasing it on the internet.
- I have learned to utilize my resources, time and energy efficiently and productively.

7. FUTURE WORK

Some of the objectives which can be achieved through this project in future are as:

- This project can be extended to support more translator systems.
- More languages support can be added to improve the efficiency calculations.
- Native translation service that depends on dictionary translations can be introduced.
- Support for more parameters for comparison like WER can be added to scale efficiency of various levels.

8. REFERENCES

The following pages and documents were referred while making this project.

- 1) Uszkoreit, Hans. "What Is Computational Linguistics?". Department of Computational Linguistics and Phonetics of Saarland, University. *[Retrieved 6 June 2020]*, Wikipedia
https://en.wikipedia.org/wiki/Computational_linguistics
- 2) Natural Language Processing, Wikipedia, *[Retrieved 20 May, 2020]*
https://en.wikipedia.org/wiki/Natural_language_processing
- 3) List of research laboratories for machine translation,
"Language Technologies Research Center". IIIT Hyderabad." Center for Indian Language Technology". IIT Bombay. Dated 14 March 2017, *[Retrieved 5 June 2020]*, Wikipedia
https://en.wikipedia.org/wiki/List_of_research_laboratories_for_machine_translation
- 4) What is Natural Language Processing? Introduction to NLP, Algorithmia, Dated 11 August 2016, *[Retrieved 10 June, 2020]*
<https://algorithmia.com/blog/introduction-natural-language-processing-nlp>
- 5) Part 5: Machine Translation Evaluation, Bonnie Dorr, *[Retrieved 20 May, 2020]*
<https://www.cs.cmu.edu/~alavie/papers/GALE-book-Ch5.pdf>
- 6) Machine Translation – 14 Current Applications and Services, Radhika Madhavan, Dated November 22, 2019, *[Retrieved 10 June 2020]*
<https://emerj.com/ai-sector-overviews/machine-translation-14-current-applications-and-services/>