İstanbul
Bilgi University

OVERVIEW OF ARABIC/TURKISH RULE-BASED VS. STATISTICAL MACHINE TRANSLATION

by

Anas ALHWID, 118200014
Kareem ALSHAWAF, 118200002

Supervised by

Assoc. Prof. Tuğba DALYAN

Submitted to the
Faculty of Engineering and Natural Sciences
in partial fulfillment of the requirements for the

Bachelor of Science

in the
Department of Computer Engineering

October, 2022

Abstract

An overview of machine translation between two languages Arabic and Turkish using two approaches; rule-based and statistical machine translation. Demonstrating the advantages and disadvantages with a comparison of the translation accuracy and data consumption of both approaches. The statistical machine translation that is considered in this paper is the IBM Model 1 with the EM Algorithm. On the other hand, for the rule-based machine translation; we have developed a model for translating verb-to-verb between Arabic and Turkish languages using language morphology for the analysis of each verb in both source and target languages to maintain the accuracy of the translation. Using our rule-based model, the user may enter a verb as infinitive or in any of the three verb tenses past, present continuous, or future, and with any of the three personal pronouns 1st person singular, 2nd person singular, or 3rd person singular, and will receive the equivalent verb with the same tense and pronoun in the target language. This paper will give a brief explanation of the IBM model 1 and the EM algorithm, and an in-depth explanation of our rule-based model from the methodology of the programming aspect and the linguistic aspect to its design, and why it was selected, in addition to experiments, results, future work, and more.

Keywords: *Statistical; IBM Model 1; EM Algorithm; Rule-based; Morphology; Machine translation; Arabic; Turkish*

Table of Contents

List of Figures

List of Abbreviations

AR	Arabic
EM	Expectation Maximization
ML	Machine Learning
MT	Machine Translation
NLP	Natural Language Processing
RBMT	Rule-Based Machine Translation
SMT	Statistical Machine Translation
TR	Turkish

1 Introduction

Machine Translation (MT) is a process of automatically translating text from one natural language (the source) to another (the target) without human involvement. To process any translation, human or automated, the meaning of a text in the original (source) language must be fully restored in the target language [1].

Rule-Based Machine Translation (RBMT) is a machine translation approach that relies on countless built-in linguistic rules and millions of bilingual dictionaries for each language pair. It requires rules that reflect the input language structure and rules that represent the output language structure, as well as monolingual and bilingual dictionaries to map input words to output words. The process mainly focus on three steps [2]:

- The grammatical input structure is analyzed by a parser.
- A preliminary representation of the message is created by the parser (different levels of abstraction are possible for this representation).
- The representation is transferred by the parser into the structure of the output language.

Statistical machine translation (SMT) is a machine translation approach that uses a large amount of bilingual data to determine which translation is most probable for a given input. Systems for statistical machine translation learn the statistical relationships between the source texts and the human translations to learn how to translate. The translation model and language model are the two most crucial parts of statistical machine translation.
In language models, the output language monolingual data is used to create the language model. Based on the translation language, the language model selects the best option from among the candidate translations.
In translation models, parallel data are used to train the statistical machine translation engine and produce a translation model. A translation model is a table with translated versions of aligned phrases, and these phrases are referred to as n-grams. The translation model's objective is to foretell potential translations for particular input texts. The process mainly focus on three steps [3]:

- The input text is separated into phrases.
- The parallel equivalents from the translation model are matched with the phrases.
- The translation's likelihood in the target language is confirmed by the language model.

2 Related Works

2.1 IBM Model 1 and the EM Algorithm

IBM Model 1 is one of the SMT systems and uses lexical translation. The benefit of this model is by directly utilizing the concept of alignments, the model enables us to recover word alignments between the source and target languages in the training set. And modern SMT systems heavily rely on the resulting alignment models. The algorithm of expectation maximization (EM) will be used to estimate the parameters of the IBM model. In statistical models for MT and other problem domains, the EM algorithm is frequently used [4] [5].

3 Methodology

3.1 Programming Language

Since this project is based on Natural language processing (NLP), Python was the perfect fit to work with as its simple syntax, and the transparent semantics of this language make it an excellent choice. In addition to enormous number of libraries and models in the field of machine learning (ML) in general and NLP in specific.

3.2 Turkish Grammatical Rules

In this project, the focus was on the positive verbs and four of their variations:

- Infinitive:
  İsim-Fiil Ekleri (-mak / -mek)

- Future Tense:
  Gelecek Zaman (-aca(k/ğ) / -ece(k/ğ))

- Present Continuous Tense:
  Şimdiki Zaman (-yor)

- Past Tense:
  Geçmiş Zaman (-(t/d)ı / -(t/d)i / -(t/d)u / -(t/d)ü)

In addition to three personal pronouns [6]:

- First Person Singular (I):
  Birinci Tekil Şahıs (Ben)

- Second Person Singular (You):
  İkinci Tekil Şahıs (Sen)

- Third Person Singular (He / She):
  Üçüncü Tekil Şahıs (O)

One of the important rules that apply in Turkish verbs is "Vowel harmony". It's used for adding or changing the vowel related to the verb tenses. It is based on the last vowel in the verb.

- "a" / "ı" Become "ı"
- "e" / "i" Become "i"
- "o" / "u" Become "u"
- "ö" / "ü" Become "ü"

3.2.1 Verb Infinitive

As shown in figure 1, creating an infinitive verb from the verb root includes adding suffix letters, and it takes three steps. First, acquiring the verb root. Second, knowing what the last vowel appears in the verb root. Third, based on the last vowel, add the correct infinitive suffix to the end of the verb. In contrast, returning the infinitive verb to the verb root takes one step which is removing the last three letters related to the infinitive verb suffix [7].

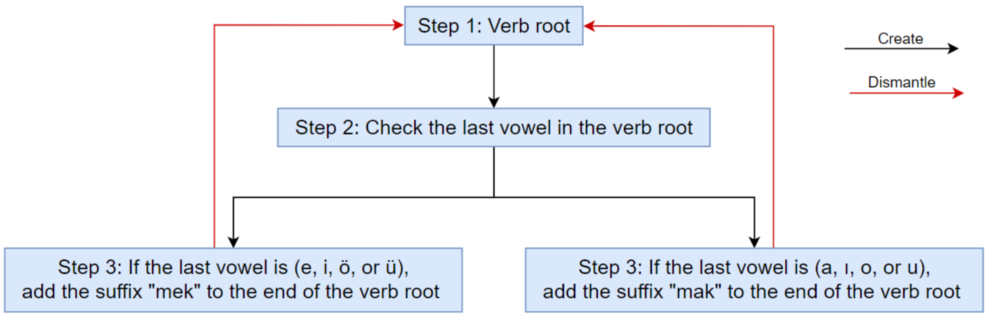
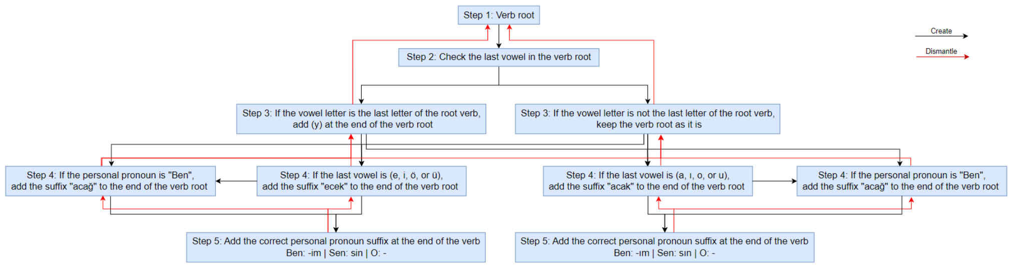

Step 1: Verb root
Step 2: Check the last vowel in the verb root
Step 3: If the last vowel is (e, i, ö, or ü), add the suffix "mek" to the end of the verb root
Step 3: If the last vowel is (a, ı, o, or u), add the suffix "mak" to the end of the verb root
Create
Dismantle

Figure 1: Verb Infinitive TR

3.2.2 Future Tense

As shown in figure 2, creating a future tense verb from the verb root takes five steps. First, acquiring the verb root. Second, knowing where the last vowel appeared in the verb root. Third, based on the position of the vowel letter in the verb root, either add an intermediate letter ”y” or not to the end of the verb root. Fourth, add the correct future tense suffix to the end of the verb. Fifth, add the correct personal pronoun suffix to the end of the verb. In contrast, returning the future tense verb to the verb root takes two or three steps. first, remove the personal pronoun suffix. Second, remove the future tense suffix. Third, only if at the end of the verb there is an intermediate letter ”y” remove it [8].



Step 1: Verb root
Create
Dismantle
Step 2: Check the last vowel in the verb root
Step 3: If the vowel letter is the last letter of the root verb, add (y) at the end of the verb root
Step 3: If the vowel letter is not the last letter of the root verb, keep the verb root as it is
Step 4: If the personal pronoun is "Ben", add the suffix "acağ" to the end of the verb root
Step 4: If the last vowel is (e, i, ö, or ü), add the suffix "ecek" to the end of the verb root
Step 4: If the last vowel is (a, ı, o, or u), add the suffix "acak" to the end of the verb root
Step 4: If the personal pronoun is "Ben", add the suffix "acağ" to the end of the verb root
Step 5: Add the correct personal pronoun suffix at the end of the verb Ben: -im | Sen: sin | O: -
Step 5: Add the correct personal pronoun suffix at the end of the verb Ben: -ım | Sen: sın | O: -

Figure 2: Future Tense TR

3.2.3 Present Continuous Tense

As shown in figure 3, creating a present continuous tense verb from the verb root takes five steps. First, acquiring the verb root. Second, knowing where the last vowel appeared in the verb root. Third, based on the position of the vowel letter in the verb, add or change the correct vowel based on the vowel harmony rule to the end of the verb root. Fourth, add the present continuous tense suffix to the end of the verb. Fifth, add the correct personal pronoun suffix to the end of the verb. In contrast, returning the present continuous tense verb to the verb root takes three or four steps. first, remove the personal pronoun suffix. Second, remove the present continuous tense suffix. Third, remove the vowel letter. Fourth, only if the letter before the last is not a vowel letter, there is no rule to follow for knowing what the verb root looks like, and what the vowel letter at the end is. Due to this issue, we get all the variants of the verb with different vowel letters at the end of the verb [9].

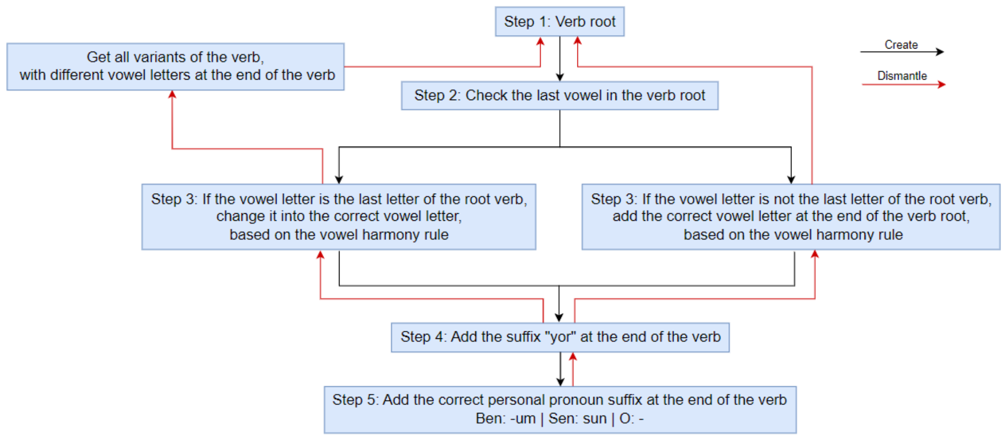

Step 1: Verb root
Create
Dismantle
Get all variants of the verb, with different vowel letters at the end of the verb
Step 2: Check the last vowel in the verb root
Step 3: If the vowel letter is the last letter of the root verb, change it into the correct vowel letter, based on the vowel harmony rule
Step 3: If the vowel letter is not the last letter of the root verb, add the correct vowel letter at the end of the verb root, based on the vowel harmony rule
Step 4: Add the suffix "yor" at the end of the verb
Step 5: Add the correct personal pronoun suffix at the end of the verb
Ben: -um | Sen: sun | O: -

Figure 3: Present Continuous Tense TR

3.2.4 Past Tense

As shown in figure 4, creating a past tense verb from the verb root takes five steps. First, acquiring the verb root. Second, knowing what the last letter appears in the verb root. Third, based on the last letter, add the correct past tense suffix to the end of the verb. Fourth, based on the last vowel that appeared in the verb, add or change the correct vowel based on the vowel harmony rule to the end of the verb. Fifth, add the correct personal pronoun suffix to the end of the verb. In contrast, returning the past tense verb to the verb root takes three or four steps. first, remove the personal pronoun suffix. Second, remove the vowel letter. Third, remove the past tense suffix. Fourth, only if the past tense suffix is ”d” there is no rule to follow for knowing what the verb root looks like, whether it has a vowel letter at the end (and which one) or it has a non. Due to this issue, we get all the variants of the verb with different vowel letters at the end of the verb and 1 without any addition [10].

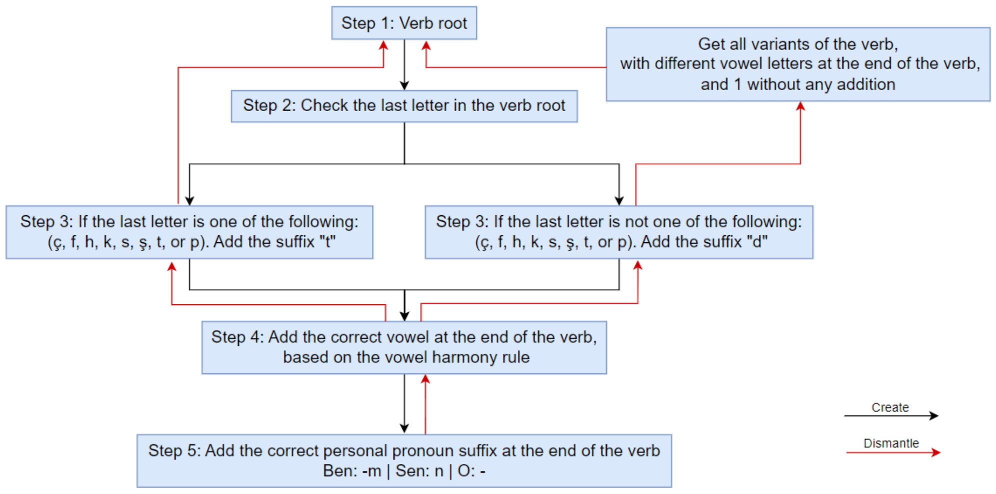

Step 1: Verb root
Get all variants of the verb, with different vowel letters at the end of the verb, and 1 without any addition
Step 2: Check the last letter in the verb root
Step 3: If the last letter is one of the following: (ç, f, h, k, s, ş, t, or p). Add the suffix "t"
Step 3: If the last letter is not one of the following: (ç, f, h, k, s, ş, t, or p). Add the suffix "d"
Step 4: Add the correct vowel at the end of the verb, based on the vowel harmony rule
Step 5: Add the correct personal pronoun suffix at the end of the verb
Ben: -m | Sen: n | O: -
Create
Dismantle

Figure 4: Past Tense TR

3.3 Arabic Grammatical Rules

In this project, the focus was on the positive verbs and four of their variations:

- Infinitive:
  مصدر الفعل

  مصادر الفعل الرباعي والخماسي والسداسي:

  (قاعدة ١: كل فعل بدأ بألف يصبح على وزن إفعال)

  (قاعدة ١.١: كل فعل بدأ بألف وقبل اخره ألف يصبح على وزن إفعالة)

  (قاعدة ١.٢: كل فعل بدأ بألف واخره ألف يصبح على وزن إفعلاء)

  (قاعدة ٢: كل فعل حرفه الثاني مشدد يصبح على وزن تفعيل)

  (قاعدة ٢.١: كل فعل حرفه الثاني مشدد واخره ألف يصبح على وزن تفعلية)

  (قاعدة ٣: كل فعل حرفه الثاني ألف يصبح على وزن مفاعلة)

  (قاعدة ٤: كل فعل بدأ بتاء يصبح على وزنه)

  (قاعدة ٤.١: كل فعل بدأ بتاء واخره ألف يصبح على وزنه ولكن تقلب الألف ياء)

  (قاعدة ٥: كل فعل لم يوفي الشروط السابقة يصبح على وزن فعللة)

- Future Tense:
  فعل المستقبل ((س) + الفعل المضارع)

- Present Continuous Tense:
  الفعل المضارع المستمر (أ - / ت - / ن - / ي)

- Past Tense:
  الفعل الماضي (- ت / - حرف منصوب)

In addition to three personal pronouns [11]:

- First Person Singular (I):
  ضمير المتكلم المفرد (أنا)

- Second Person Singular (You):
  ضمير المخاطَب المفرد (أنتَ / أنتِ)

- Third Person Singular (He / She):
  ضمير الغائب المفرد (هو / هي)

3.3.1 Verb Infinitive

As shown in figure 5, creating an infinitive verb from the verb root includes adding or changing prefix and/or middle and/or suffix letter(s), and it takes two steps. First, acquiring the verb root. Second, in the Arabic language, there are different grammatical rules for turning the verb root into an infinitive verb. These rules are divided into two main categories:

- Three-letter verb root: The grammatical rules are based on the meaning of the verb.
- More than three-letter verb root: The grammatical rules are set to five rules with special case(s) in some of them.

In this project, we have considered only the verb root with more than three-letter. In contrast, returning the infinitive verb to the verb root takes one step which is removing the prefix and/or middle and/or suffix letter(s) related to the infinitive verb [12] [13].

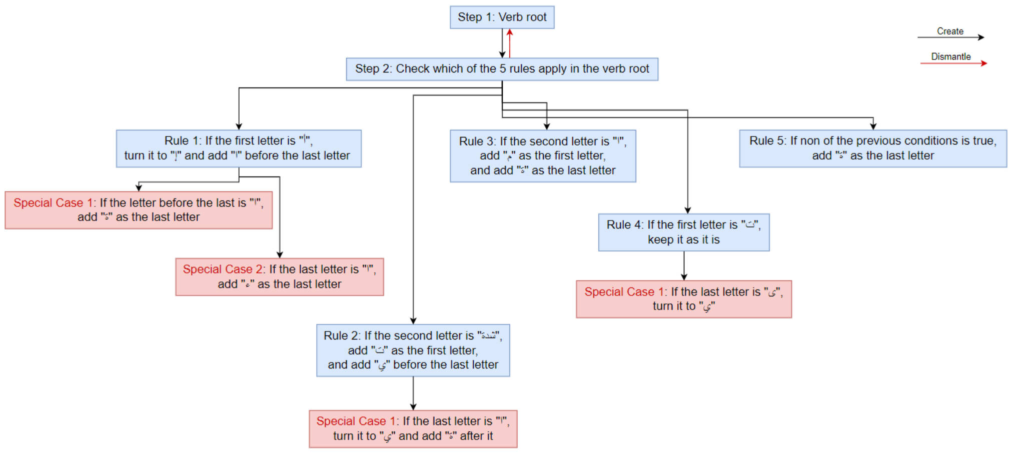

Step 1: Verb root
Create
Dismantle
Step 2: Check which of the 5 rules apply in the verb root
Rule 1: If the first letter is "أ", turn it to "إ" and add "ا" before the last letter
Rule 3: If the second letter is "ا", add "م" as the first letter, and add "ة" as the last letter
Rule 5: If non of the previous conditions is true, add "ة" as the last letter
Special Case 1: If the letter before the last is "ا", add "ة" as the last letter
Rule 4: If the first letter is "ت", keep it as it is
Special Case 2: If the last letter is "ا", add "ء" as the last letter
Special Case 1: If the last letter is "ى", turn it to "ي"
Rule 2: If the second letter is "شدة", add "ت" as the first letter, and add "ي" before the last letter
Special Case 1: If the last letter is "ا", turn it to "ي" and add "ة" after it

Figure 5: Verb Infinitive AR

3.3.2 Future Tense

As shown in figure 6, creating a future tense verb from the verb root takes four steps. First, acquiring the verb root. Second, knowing what is the personal pronoun. Third, based on the personal pronoun, add the correct present continuous tense prefix, and in some special cases for specific personal pronouns, add their suffix. Fourth, add the future tense prefix to the beginning of the verb. In contrast, returning the future tense verb to the verb root takes two steps. First, remove the future tense prefix. Second, remove the present continuous tense prefix and if there is any suffix related to a specific personal pronoun [14].

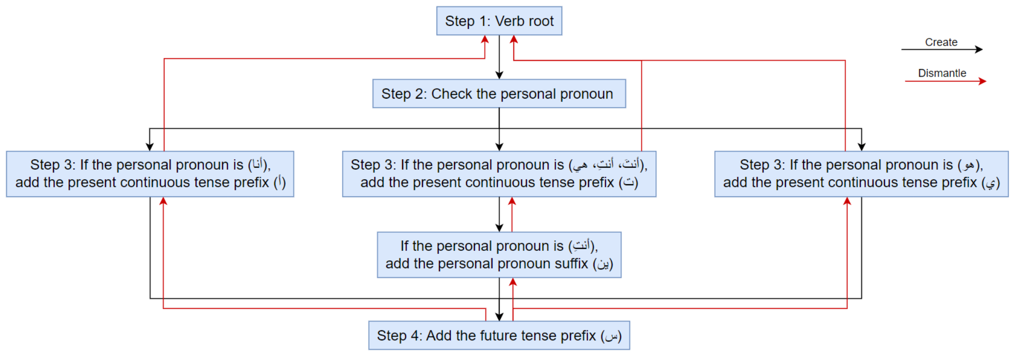

Step 1: Verb root
Create
Dismantle
Step 2: Check the personal pronoun
Step 3: If the personal pronoun is (أنا), add the present continuous tense prefix (أ)
Step 3: If the personal pronoun is (أنتَ، أنتِ، هي), add the present continuous tense prefix (ت)
Step 3: If the personal pronoun is (هو), add the present continuous tense prefix (ي)
If the personal pronoun is (أنتِ), add the personal pronoun suffix (ين)
Step 4: Add the future tense prefix (س)

Figure 6: Future Tense AR

3.3.3 Present Continuous Tense

As shown in figure 7, creating a present continuous tense verb from the verb root takes three steps. First, acquiring the verb root. Second, knowing what is the personal pronoun. Third, based on the personal pronoun, add the correct present continuous tense prefix, and in some special cases for specific personal pronouns, add their suffix. In contrast, returning the present continuous tense verb to the verb root takes one step. Remove the present continuous tense prefix and if there is any suffix related to a specific personal pronoun [15].

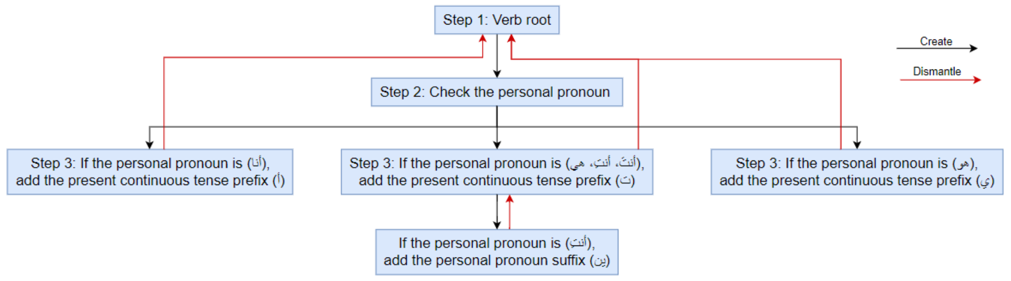

Step 1: Verb root
Create
Dismantle
Step 2: Check the personal pronoun
Step 3: If the personal pronoun is (أنا), add the present continuous tense prefix (أ)
Step 3: If the personal pronoun is (أنتَ، أنتِ، هي), add the present continuous tense prefix (ت)
Step 3: If the personal pronoun is (هو), add the present continuous tense prefix (ي)
If the personal pronoun is (أنتِ), add the personal pronoun suffix (ين)

Figure 7: Present Continuous Tense AR

3.3.4 Past Tense

As shown in figure 8, creating a past tense verb from the verb root takes three steps. First, acquiring the verb root. Second, knowing what is the personal pronoun. Third, based on the personal pronoun, add the correct past tense suffix. In contrast, returning the past tense verb to the verb root takes one step. Remove the past tense suffix [16].

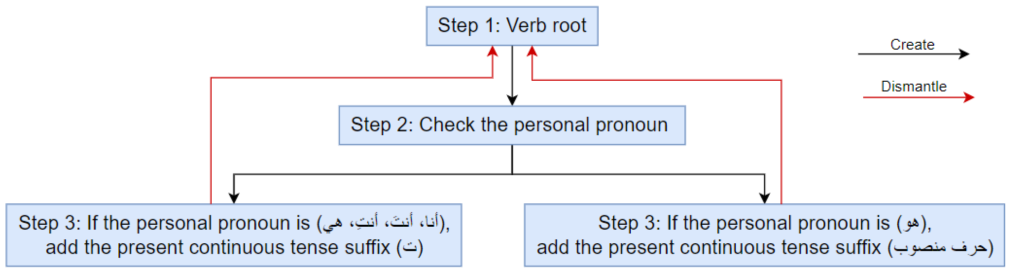

Step 1: Verb root
Create
Dismantle
Step 2: Check the personal pronoun
Step 3: If the personal pronoun is (أنا، أنتَ، أنتِ، هي), add the present continuous tense suffix (ت)
Step 3: If the personal pronoun is (هو), add the present continuous tense suffix (حرف منصوب)

Figure 8: Past Tense AR

4 Design of the Project

4.1 Source to Target Language Translation Flowchart

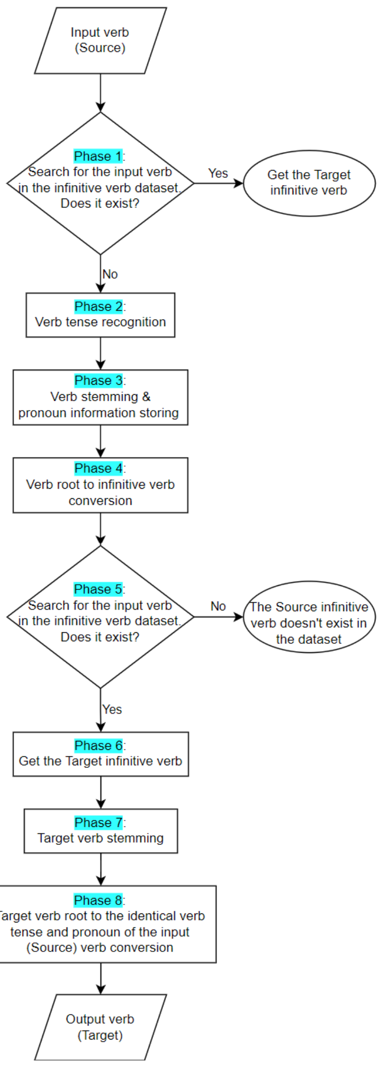

Input verb (Source)
Phase 1: Search for the input verb in the infinitive verb dataset Does it exist?
Yes
Get the Target infinitive verb
No
Phase 2: Verb tense recognition
Phase 3: Verb stemming & pronoun information storing
Phase 4: Verb root to infinitive verb conversion
Phase 5: Search for the input verb in the infinitive verb dataset Does it exist?
No
The Source infinitive verb doesn't exist in the dataset
Yes
Phase 6: Get the Target infinitive verb
Phase 7: Target verb stemming
Phase 8: Target verb root to the identical verb tense and pronoun of the input (Source) verb conversion
Output verb (Target)

Figure 9: Source to Target Translation Flowchart

5 Design Methodology

The project uses the Agile model as its main methodology for various reasons including its suitability with our vision and plan. The agile model approach makes the iterative development of the project easy and clear.

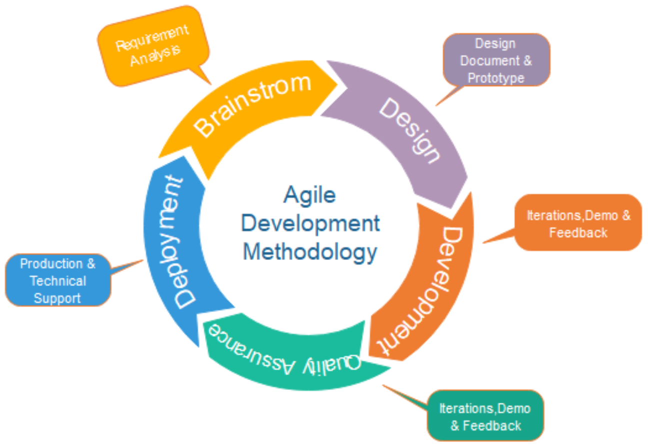

Requirement Analysis
Brainstrom
Design Document & Prototype
Design
Agile Development Methodology
Iterations,Demo & Feedback
Development
Production & Technical Support
Deployment
Quality Assurance
Iterations,Demo & Feedback

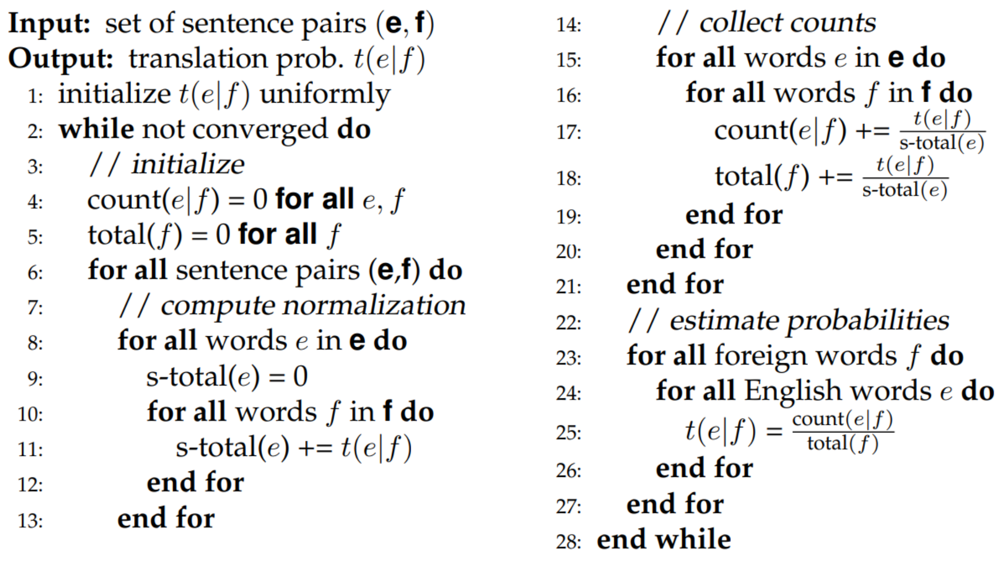
Figure 10: Agile Model

The agile methods divide tasks into smaller iterations or pieces and do not involve long-term planning directly. The project scope and requirements are established at the start of the development process [17].

- Requirements gathering phase: This phase consists of searching and gathering information on the subject of the project from similar applications, the best programming libraries for NLP and MT, the best model, and much other information related to MT.
- Design the requirements phase: The process begins with listing all the requirements needed for the project from the information collected from the previous stage (Requirements gathering). Then, the information gets sorted into different stages for the different parts of the development such as coding, and linguistics.
- Construction/iteration phase: The implementation part of the previous phase (Design the requirements). All coding and linguistics parts will be applied and implemented. The project will undergo various stages of improvement, so it includes simple and minimal functionality.
- Testing phase: After the code has been implemented, the testing stage begins to ensure that the translation runs correctly and does not include any coding bugs or linguistical issues.
- Deployment phase: Once everything has been finalized, the product is ready for release.
- Feedback phase: The final phase after releasing the product is feedback. In this phase, the team receives feedback on the project and works through it.

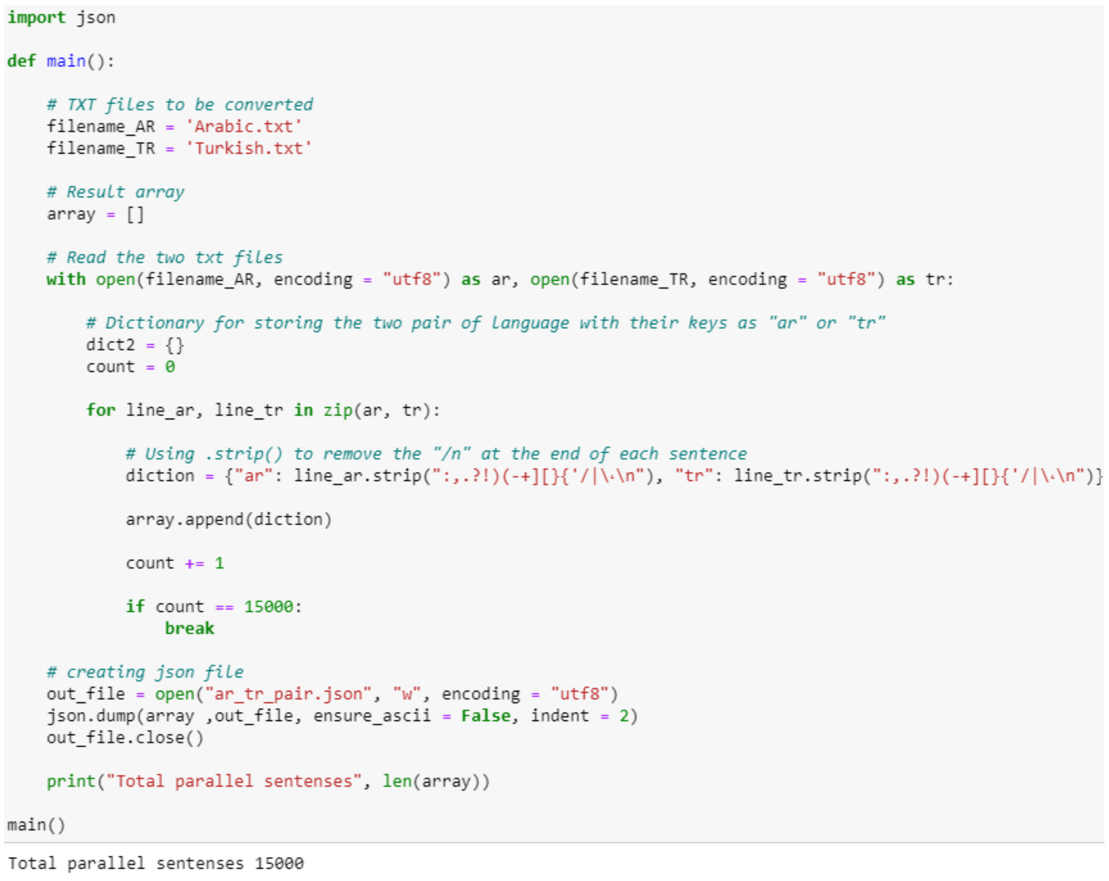
6 Experiments

Starting the idea of creating a translation program between Arabic and Turkish was on SMT using IBM Model 1 and the EM Algorithm as shown in figure 11 [5]. We have collected more than ten thousand sentences from TED Talks between the two languages Arabic and Turkish and paralleled them as shown in figure 12 [18].



```
Input: set of sentence pairs $(\mathbf{e}, \mathbf{f})$
Output: translation prob. $t(e|f)$
initialize $t(e|f)$ uniformly
while not converged do
    // initialize
    count$(e|f)$ = 0 for all $e, f$
    total$(f)$ = 0 for all $f$
    for all sentence pairs $(\mathbf{e},\mathbf{f})$ do
        // compute normalization
        for all words $e$ in $\mathbf{e}$ do
            s-total$(e)$ = 0
            for all words $f$ in $\mathbf{f}$ do
                s-total$(e)$ += $t(e|f)$
            end for
        end for
        // collect counts
        for all words $e$ in $\mathbf{e}$ do
            for all words $f$ in $\mathbf{f}$ do
                count$(e|f)$ += $\frac{t(e|f)}{\text{s-total}(e)}$
                total$(f)$ += $\frac{t(e|f)}{\text{s-total}(e)}$
            end for
        end for
    end for
    // estimate probabilities
    for all foreign words $f$ do
        for all English words $e$ do
            $t(e|f) = \frac{\text{count}(e|f)}{\text{total}(f)}$
        end for
    end for
end while
```

Figure 11: IBM Model 1 and EM: Pseudocode



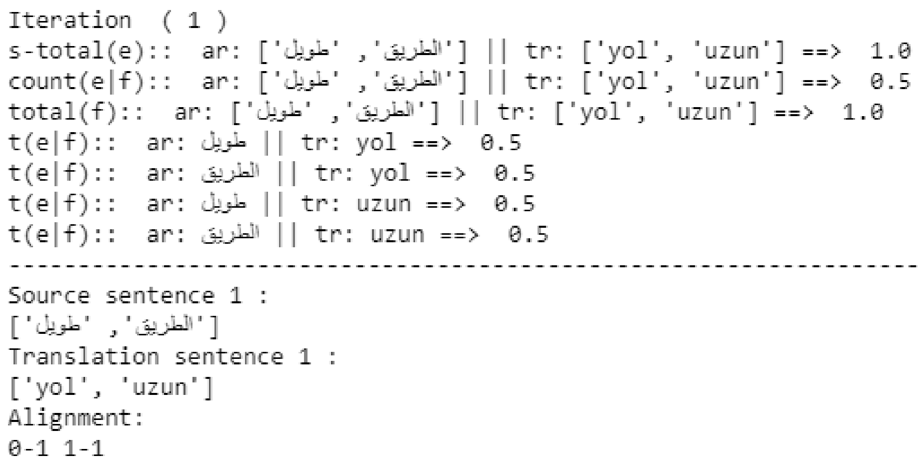
```
import json

def main():

    # TXT files to be converted
    filename_AR = 'Arabic.txt'
    filename_TR = 'Turkish.txt'

    # Result array
    array = []

    # Read the two txt files
    with open(filename_AR, encoding = "utf8") as ar, open(filename_TR, encoding = "utf8") as tr:

        # Dictionary for storing the two pair of language with their keys as "ar" or "tr"
        dict2 = {}
        count = 0

        for line_ar, line_tr in zip(ar, tr):

            # Using .strip() to remove the "/n" at the end of each sentence
            diction = {"ar": line_ar.strip(":;,.?!)(-+][]{'/|\·\n"), "tr": line_tr.strip(":;,.?!)(-+][]{'/|\·\n")}

            array.append(diction)

            count += 1

            if count == 15000:
                break

    # creating json file
    out_file = open("ar_tr_pair.json", "w", encoding = "utf8")
    json.dump(array ,out_file, ensure_ascii = False, indent = 2)
    out_file.close()

    print("Total parallel sentenses", len(array))

main()
```

```
Total parallel sentenses 15000
```

Figure 12: Text files to JSON file conversion

The advantage of the SMT models is that they give great results in word alignment in pairs of sentences from different languages using the probability of likelihood between every two words and iterating that will improve the probability. On the contrary, the disadvantage is that they require large data to do so. The more data they consume, the better result they provide.

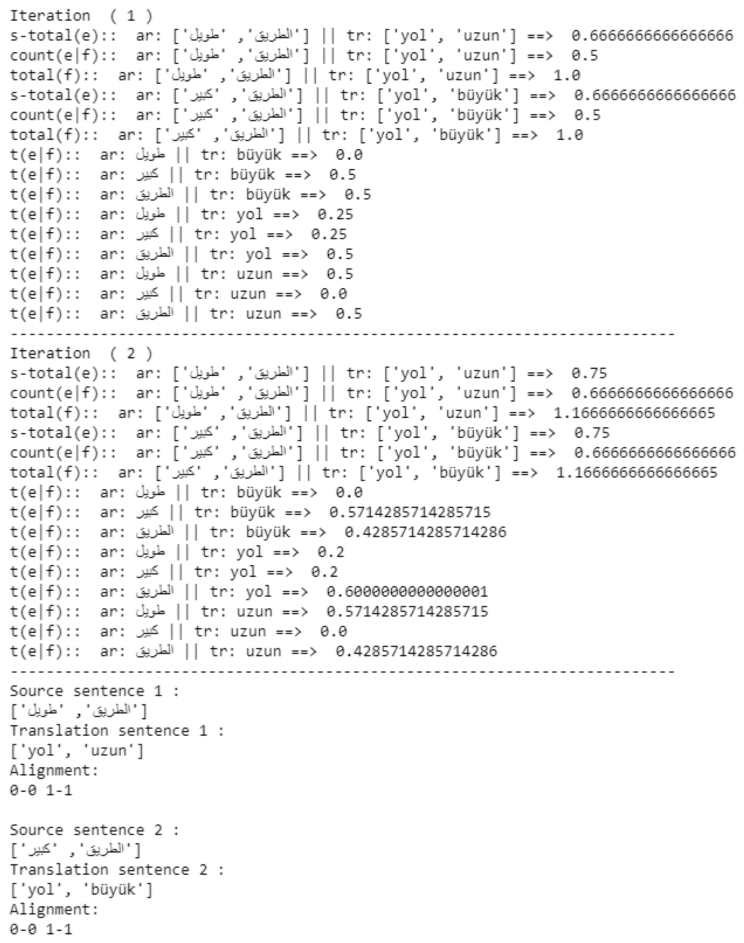
As shown in figure 13, assuming we have only one pair of sentences and each sentence contains two words. In this case, the number of iterations will not matter as the probability of each word with each of the corresponding two words from the other language will start at 0.5 and will stay at this percentage. The alignment can also be noticed as incorrect since its aligning the first word in Arabic with the second in Turkish instead of the first in Turkish.



```
Iteration  ( 1 )
s-total(e)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  1.0
count(e|f)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  0.5
total(f)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  1.0
t(e|f)::  ar: طويل || tr: yol ==>  0.5
t(e|f)::  ar: الطريق || tr: yol ==>  0.5
t(e|f)::  ar: طويل || tr: uzun ==>  0.5
t(e|f)::  ar: الطريق || tr: uzun ==>  0.5
-----------------------------------------------------------------
Source sentence 1 :
['الطريق', 'طويل']
Translation sentence 1 :
['yol', 'uzun']
Alignment:
0-1 1-1
```

Figure 13: IBM 1 With EM Algorithm: Arabic/Turkish Translation Alignment

As shown in figure 14, when we add one more pair of sentences and each sentence contains two words with one repeated word from the previous sentence. In this case, the number of iterations will matter as the probability of each word with each of the corresponding four words from the other language will start at 0.5 and with each iteration will either increase or decrease. The alignment can also be noticed as correct since its aligning the first word in Arabic with the first in Turkish and the second word in Arabic with the second in Turkish in both pairs of sentences.

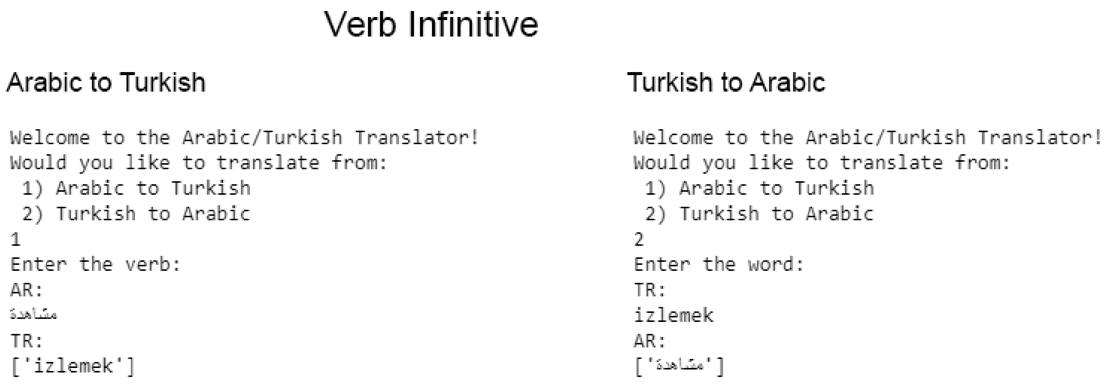


```
Iteration  ( 1 )
s-total(e)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  0.6666666666666666
count(e|f)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  0.5
total(f)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  1.0
s-total(e)::  ar: ['الطريق', 'كبير'] || tr: ['yol', 'büyük'] ==>  0.6666666666666666
count(e|f)::  ar: ['الطريق', 'كبير'] || tr: ['yol', 'büyük'] ==>  0.5
total(f)::  ar: ['الطريق', 'كبير'] || tr: ['yol', 'büyük'] ==>  1.0
t(e|f)::  ar: طويل || tr: büyük ==>  0.0
t(e|f)::  ar: كبير || tr: büyük ==>  0.5
t(e|f)::  ar: الطريق || tr: büyük ==>  0.5
t(e|f)::  ar: طويل || tr: yol ==>  0.25
t(e|f)::  ar: كبير || tr: yol ==>  0.25
t(e|f)::  ar: الطريق || tr: yol ==>  0.5
t(e|f)::  ar: طويل || tr: uzun ==>  0.5
t(e|f)::  ar: كبير || tr: uzun ==>  0.0
t(e|f)::  ar: الطريق || tr: uzun ==>  0.5
-----------------------------------------------------------------------
Iteration  ( 2 )
s-total(e)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  0.75
count(e|f)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  0.6666666666666666
total(f)::  ar: ['الطريق', 'طويل'] || tr: ['yol', 'uzun'] ==>  1.1666666666666665
s-total(e)::  ar: ['الطريق', 'كبير'] || tr: ['yol', 'büyük'] ==>  0.75
count(e|f)::  ar: ['الطريق', 'كبير'] || tr: ['yol', 'büyük'] ==>  0.6666666666666666
total(f)::  ar: ['الطريق', 'كبير'] || tr: ['yol', 'büyük'] ==>  1.1666666666666665
t(e|f)::  ar: طويل || tr: büyük ==>  0.0
t(e|f)::  ar: كبير || tr: büyük ==>  0.5714285714285715
t(e|f)::  ar: الطريق || tr: büyük ==>  0.4285714285714286
t(e|f)::  ar: طويل || tr: yol ==>  0.2
t(e|f)::  ar: كبير || tr: yol ==>  0.2
t(e|f)::  ar: الطريق || tr: yol ==>  0.6000000000000001
t(e|f)::  ar: طويل || tr: uzun ==>  0.5714285714285715
t(e|f)::  ar: كبير || tr: uzun ==>  0.0
t(e|f)::  ar: الطريق || tr: uzun ==>  0.4285714285714286
-----------------------------------------------------------------------
Source sentence 1 :
['الطريق', 'طويل']
Translation sentence 1 :
['yol', 'uzun']
Alignment:
0-0 1-1

Source sentence 2 :
['الطريق', 'كبير']
Translation sentence 2 :
['yol', 'büyük']
Alignment:
0-0 1-1
```

Figure 14: IBM 1 With EM Algorithm: Arabic/Turkish Translation Alignment

7 Results and Discussion

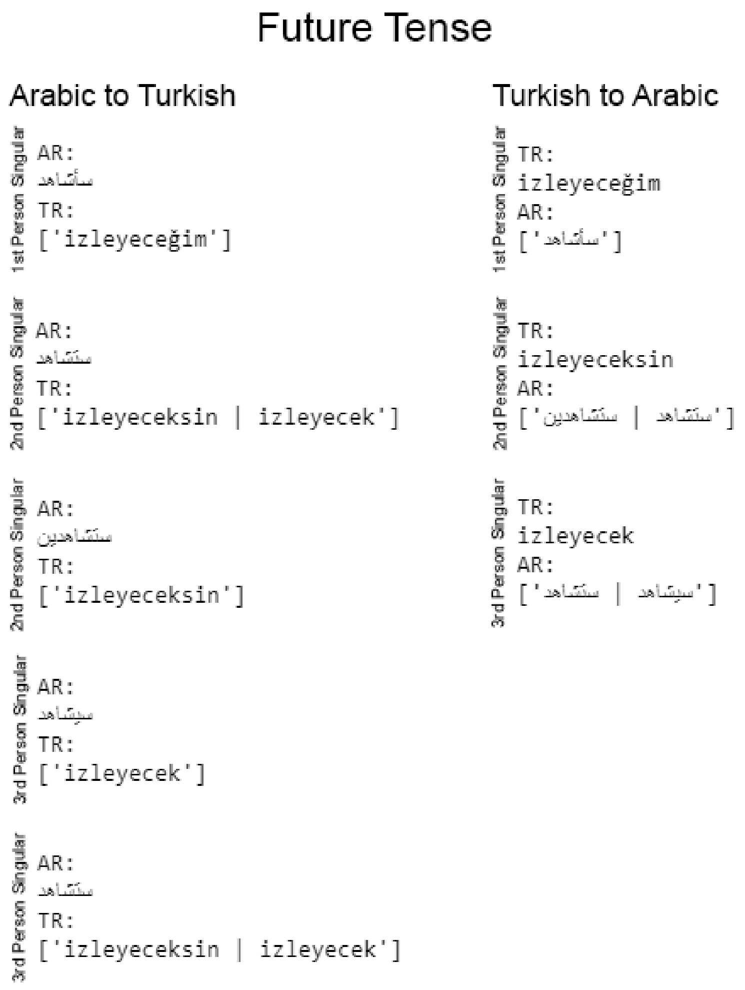
The following figures 15 16 17 18 are a demonstration of the translation results of our rule-based model from Arabic to Turkish and vice versa. These results are showing all the grammatical rules applied in both languages, Arabic and Turkish.



```
Welcome to the Arabic/Turkish Translator!
Would you like to translate from:
 1) Arabic to Turkish
 2) Turkish to Arabic
1
Enter the verb:
AR:
مشاهدة
TR:
['izlemek']
```

```
Welcome to the Arabic/Turkish Translator!
Would you like to translate from:
 1) Arabic to Turkish
 2) Turkish to Arabic
2
Enter the word:
TR:
izlemek
AR:
['مشاهدة']
```

Figure 15: Rule Based: Verb Infinitive



Future Tense

Arabic to Turkish

1st Person Singular
```
AR:
سأشاهد
TR:
['izleyeceğim']
```

2nd Person Singular
```
AR:
ستشاهد
TR:
['izleyeceksin | izleyecek']
```

2nd Person Singular
```
AR:
ستشاهدين
TR:
['izleyeceksin']
```

3rd Person Singular
```
AR:
سيشاهد
TR:
['izleyecek']
```

3rd Person Singular
```
AR:
ستشاهد
TR:
['izleyeceksin | izleyecek']
```

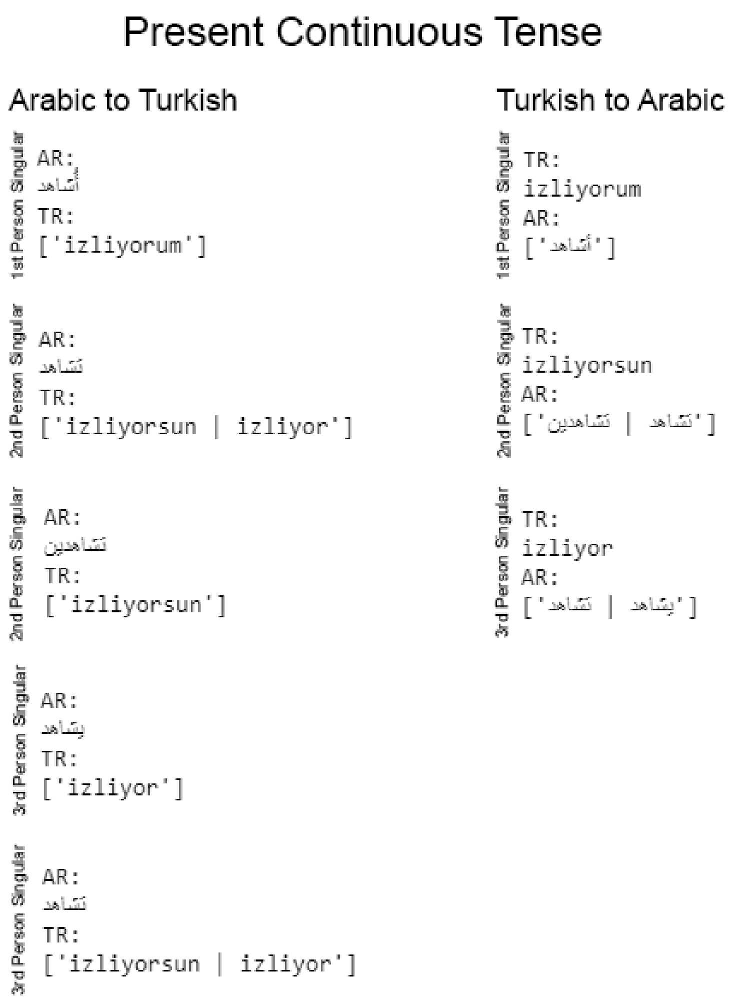
Turkish to Arabic

1st Person Singular
```
TR:
izleyeceğim
AR:
['سأشاهد']
```

2nd Person Singular
```
TR:
izleyeceksin
AR:
['ستشاهد | ستشاهدين']
```

3rd Person Singular
```
TR:
izleyecek
AR:
['سيشاهد | ستشاهد']
```

Figure 16: Rule Based: Future Tense



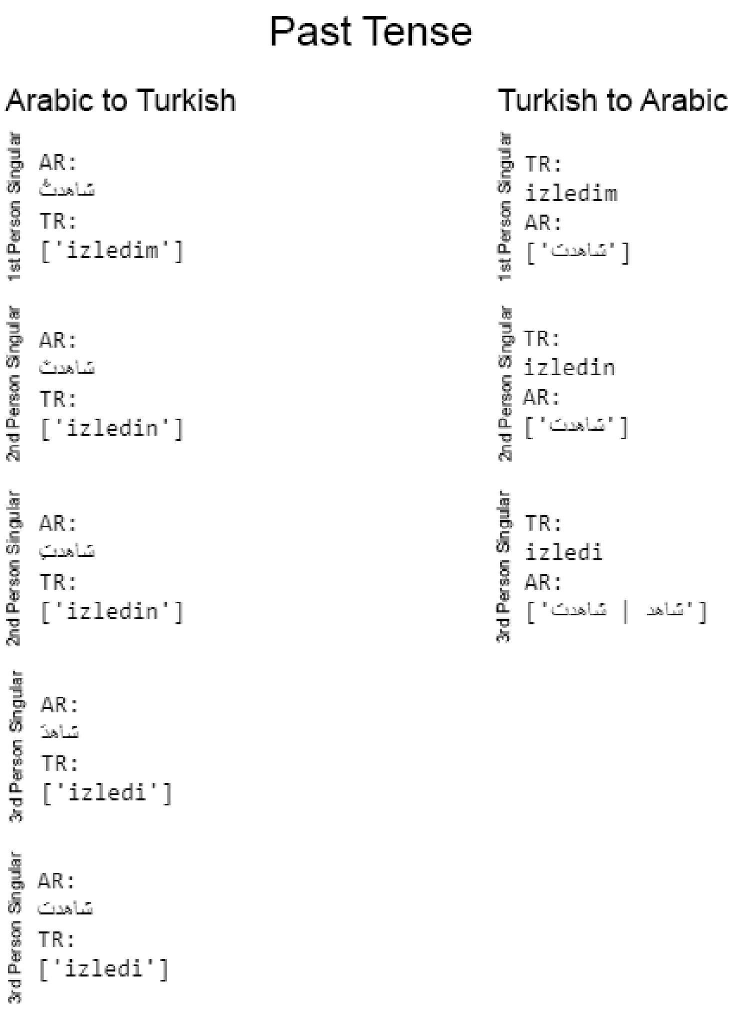
Arabic to Turkish

```
AR:
أشاهد
TR:
['izliyorum']

AR:
تشاهد
TR:
['izliyorsun | izliyor']

AR:
تشاهدين
TR:
['izliyorsun']

AR:
يشاهد
TR:
['izliyor']

AR:
تشاهد
TR:
['izliyorsun | izliyor']
```

Turkish to Arabic

```
TR:
izliyorum
AR:
['أشاهد']

TR:
izliyorsun
AR:
['تشاهد | تشاهدين']

TR:
izliyor
AR:
['يشاهد | تشاهد']
```

Figure 17: Rule Based: Present Continuous Tense



Past Tense

Arabic to Turkish

```
1st Person Singular
AR:
شاهدتُ
TR:
['izledim']

2nd Person Singular
AR:
شاهدتَ
TR:
['izledin']

2nd Person Singular
AR:
شاهدتِ
TR:
['izledin']

3rd Person Singular
AR:
شاهدَ
TR:
['izledi']

3rd Person Singular
AR:
شاهدت
TR:
['izledi']
```

Turkish to Arabic

```
1st Person Singular
TR:
izledim
AR:
['شاهدت']

2nd Person Singular
TR:
izledin
AR:
['شاهدت']

3rd Person Singular
TR:
izledi
AR:
['شاهد | شاهدت']
```

Figure 18: Rule Based: Past Tense

8 Realistic Constrains

8.1 Social, Environmental and Economical Impact

For the social impact, the project will provide one of the most important aspects of social life which is connection. Connecting people speaking different languages using the best translation.

For the environmental impact, the project is using the rule-based approach which is designed for less data consumption and fast compiling which would help to reduce electric waste.

For the economical impact, the RBMT will be provided for public use free of charge. Therefore, the application can be used by users all around the globe.

8.2 Cost Analysis

This project is highly costly since it requires people from different fields in computer engineering and linguistics for both Arabic and Turkish languages. In addition, in case this project is published on the internet, it would require the cost of servers and maintenance.

8.3 Standards

The aim of the project is to offer translation between Arabic and Turkish to the users. Users can use the application on their devices, and a global version on the server may be run by the main developers. Our standard is very clear and straightforward, we will not provide false or misleading translations. The main purpose is the integrity of the translation, and we take that seriously.

8.3.1 NSPE Code of Ethics

The project follows NSPE's code of ethics, including but not limited to:

- Engineers shall avoid deceptive acts.
- Engineers shall be guided in all their relations by the highest standards of honesty and integrity.
- Engineers shall not disclose, without consent, confidential information concerning the business affairs or technical processes of any present or former client or employer, or public body on which they serve.
- Engineers shall act for each employer or client as faithful agents or trustees.

8.3.2 IEEE Standards

- IEEE/ISO/IEC P23026 - ISO/IEC/IEEE International Standard - Systems and Software Engineering – Engineering and Management of Websites for Systems, Software, and Services Information [19].
  - This document defines system engineering and management requirements for the life cycle of websites including strategy, design, engineering, testing and validation, and management and sustainment for Intranet and Extranet environments.
  - This document applies to those using web technology to present information and communications technology (ICT) information, such as information for users of systems and services, plans and reports for systems and software engineering projects.
- IEEE 15026-2-2011 - IEEE Standard–Adoption of ISO/IEC 15026-2:2011 Systems and Software Engineering–Systems and Software Assurance–Part 2: Assurance Case [20].
  - ISO/IEC 15026-2:2011 specifies minimum requirements for the structure and contents of an assurance case to improve the consistency and comparability of assurance cases and to facilitate stakeholder communications, engineering decisions, and other uses of assurance cases.

9 Risk Analysis and Changes

The rule-based MT offers great translation for verb-to-verb translation, but as it grows larger to work with sentences, it will face many risks which will require fast and efficient solutions.

- Risk 1: Misspelled input: Users may misspell while writing the words which would cause either wrong translation or unavailable translation.
- Change: Provide the model with a spell correction for reducing spelling inaccuracy.
- Risk 2: Rule-based limitation: The Rule-based approach will come to a point where it will face limitations that would require other approaches to be used.
- Change: Both statistical and Neural MT approaches can be used alongside the Rule-based in case of any limitations.
- Risk 3: Overwhelming Costs: As the project grows, it will require very high grammatical resources from the linguists and a lot of programming from the developers which will become very expensive.
- Change: We aim to keep the main idea on a rule-based approach with integrating a statistical approach that would lower the effort and cost of developing a fully rule-based approach.

10 Challenges

- First: Since we have very basic knowledge of the Turkish language it was a difficult challenge to learn a new language.
- Second: Understanding the grammatical rules in both Arabic and Turkish, then creating our design to be applied as a Python code.
- Third: The process of changing the verb root to some verb tenses was not the same contrariwise which made it a bit tricky to be designed.

11 Future Work

The current version of the Arabic/Turkish Translation program includes all the functions that were mapped during the first stages. However, this project is massively big and there are a lot of development to be done. Future work could include:

- Include all the grammatical rules for the Arabic language.
- Include all the grammatical rules for the Turkish language.
- Include translation for more than only one word with alignment.
- Include SMT and apply some feature to the program such as, next word suggestion.
- Increase the language pair dataset.
- Develop spell correction for the Arabic language.
- Develop spell correction for the Turkish language.

12 Conclusion

Connection is an essential aspect of our daily lives. This project will provide users with the ability to translate between Arabic and Turkish without worrying about the language barrier. What makes this project stand against other popular translation applications is its usage of lesser data to be consumed, fast translation, and providing accurate translation.

References

[1] SYSTRAN, *What is machine translation? rule based machine translation vs. statistical machine translation*, 2022. [Online]. Available: https://www.systransoft.com/systran/translation-technology/what-is-machine-translation/ (visited on 09/25/2022).

[2] C. Yalangozian, *Rule-based machine translation*, 2022. [Online]. Available: https://machinetranslate.org/rule-based-machine-translation (visited on 09/25/2022).

[3] ——, *Statistical machine translation*, 2022. [Online]. Available: https://machinetranslate.org/statistical-machine-translation (visited on 09/25/2022).

[4] M. Collins, *Statistical machine translation: Ibm models 1 and 2*, 2022. [Online]. Available: https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1162/handouts/Collins_annotated.pdf (visited on 09/25/2022).

[5] P. Koehn, *Ibm model 1 and the em algorithm*, 2022. [Online]. Available: http://mt-class.org/jhu/slides/lecture-ibm-model1.pdf (visited on 09/25/2022).

[6] tr.wikipedia.org, *Şahıs eki*, 2022. [Online]. Available: https://tr.wikipedia.org/wiki/%C5%9Eah%C4%B1s_eki (visited on 09/25/2022).

[7] wikibooks, *Turkish/infinitive*, 2022. [Online]. Available: https://en.wikibooks.org/wiki/Turkish/Infinitive (visited on 09/25/2022).

[8] turkishbasics, *Future tense*, 2022. [Online]. Available: http://turkishbasics.com/verbs/future-tense.php (visited on 09/25/2022).

[9] ——, *Present continuous tense*, 2022. [Online]. Available: http://turkishbasics.com/verbs/present-continuous-tense.php (visited on 09/25/2022).

[10] ——, *Definite past tense*, 2022. [Online]. Available: http://turkishbasics.com/verbs/definite-past-tense.php (visited on 09/25/2022).

[11] ar.wikipedia.org/, *Arabic pronouns*, 2022. [Online]. Available: https://ar.wikipedia.org/wiki/%D8%B6%D9%85%D8%A7%D8%A6%D8%B1_%D8%A7%D9%84%D9%84%D8%BA%D8%A9_%D8%A7%D9%84%D8%B9%D8%B1%D8%A8%D9%8A%D8%A9 (visited on 09/25/2022).

[12] analbahr, *What is the infinitive in the arabic language: Definition, parsing, clear examples*, 2019. [Online]. Available: https://analbahr.com/%D9%85%D8%A7-%D9%87%D9%88-%D8%A7%D9%84%D9%85%D8%B5%D8%AF%D8%B1-%D9%81%D9%8A-%D8%A7%D9%84%D9%84%D8%BA%D8%A9-%D8%A7%D9%84%D8%B9%D8%B1%D8%A8%D9%8A%D8%A9-%D8%9F-%D8%AA%D8%B9%D8%B1%D9%8A%D9%81-%D8%8C/ (visited on 09/25/2022).

[13] nouronkafaf, *Infinitive*, 2022. [Online]. Available: https://sites.google.com/site/nouronkafaf/home/masdar (visited on 09/25/2022).

[14] M. Qawakzah, *The future tense in the arabic language*, 2013. [Online]. Available: http://mohamedrabeea.net/library/pdf/91fa3884-ab27-4cc7-802a-e50772337058.pdf (visited on 09/25/2022).

[15] H. A. Bakri, *Definition of the present tense*, 2018. [Online]. Available: https://mawdoo3.com/%D8%AA%D8%B9%D8%B1%D9%8A%D9%81_%D8%A7%D9%84%D9%81%D8%B9%D9%84_%D8%A7%D9%84%D9%85%D8%B6%D8%A7%D8%B1%D8%B9 (visited on 09/25/2022).

[16] S. K. A. Zar, *Definition of the past tense*, 2020. [Online]. Available: https://mawdoo3.com/%D8%AA%D8%B9%D8%B1%D9%8A%D9%81_%D8%A7%D9%84%D9%81%D8%B9%D9%84_%D8%A7%D9%84%D9%85%D8%A7%D8%B6%D9%8A (visited on 09/25/2022).

[17] javatpoint, *Agile model*, 2022. [Online]. Available: https://www.javatpoint.com/software-engineering-agile-model#:~:text=%22Agile%20process%20model%22%20refers%20to,beginning%20of%20the%20development%20process. (visited on 09/25/2022).

[18] ajinkyakulkarni14, *Ted-multilingual-parallel-corpus*, 2016. [Online]. Available: https://github.com/ajinkyakulkarni14/TED-Multilingual-Parallel-Corpus (visited on 09/25/2022).

[19] I. SA, *Iso/iec/ieee international standard - systems and software engineering – engineering and management of websites for systems, software, and services information*, 2020. [Online]. Available: https://standards.ieee.org/ieee/23026/10425/ (visited on 06/27/2022).

[20] ISO.org, *Iso/iec 15026-2:2011 systems and software engineering — systems and software assurance — part 2: Assurance case*, 2011. [Online]. Available: https://www.iso.org/standard/52926.html (visited on 06/27/2022).