RULE-WEIGHT LEARNING FOR KAZAKH-TURKISH MACHINE TRANSLATION

by

SEWALE MUSADAQ TAHA

Electrical and Computer Engineering

Submitted to the Graduate School of Science and Engineering

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

ALTINBAŞ UNIVERSITY

2020

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Asst. Prof. Dr. Francis M. TYERS	Asst. Prof. Dr. Sefer KURNAZ
Co-Supervisor	Supervisor

Examining Committee Members:

Asst. Prof. Dr. Sefer KURNAZ	School of Engineering and Natural Sciences, Altinbas University	
Asst. Prof. Dr. Francis M. TYERS	Department of Linguistics, Indiana University	
Prof. Dr. Osman Nuri UÇAN	School of Engineering and Natural Sciences, Altinbas University	
Prof. Dr. Adem KARAHOCA	Faculty of Engineering and Architecture, Computer Engineering Department, Nisantasi University	
Asst. Prof. Dr. Abdullahi Abdu IBRAHIM	School of Engineering and Natural Sciences, Altinbas University	
Asst. Prof. Dr. Oğuz KARAN	School of Engineering and Natural Sciences, Software Engineering Department, Altinbas University	

Asst. Prof. Dr. Zeynep ALTAN Faculty of Engineering and Architecture,

Software Engineering Department,

Beykent University ___________________

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy.

____

Asst. Prof. Çağatay AYDIN

Head of Department

Approval Date of Graduate School of

Science and Engineering: ____/____/____

____

Prof. Oğuz BAYAT

Director

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis. This thesis has also not been submitted for any degree in any university previously.

SEWALE MUSADAQ TAHA

ACKNOWLEDGEMENTS

I wish to express my sincere regards to my supervisor Dr. Sefer Kurnaz for the continuous support of my Ph.D. study and research. Besides my supervisor, I would like to pay my special regards to my Co-Supervisor Dr. Francis M. Tyers who has the substance of a genius: he convincingly guided and encouraged me to be professional and aspiring guidance, invaluably constructive criticism and friendly advice during the project work.

My sincere thanks also goes to the rest the rest of my thesis committee: Dr. Jonathan N. Washington, Dr. Oğuz karan for insightful comments, and hard questions.

I thank my fellow in Apertium organization, for stimulating discussions and answering my questions at any moment. I also thank my friends Aboelhamd Aly, Onur Yasin Yücel and Dr. Selma Özaydın for their help and support.

Last but not the least, I wish to acknowledge the support and great love of my family, my parents, Nadia and Musaddak; my sister, Şenay; and my aunt, Bahar. They kept me going on and this work would not have been possible without their input.

ABSTRACT

RULE-WEIGHT LEARNING FOR KAZAKH-TURKISH MACHINE TRANSLATION

TAHA SEWALE MUSADAQ

PhD, Electrical and Computer Engineering, Altınbaş University,

Supervisor: Asst. Prof. Dr. Sefer KURNAZ

Co-Supervisor: Asst. Prof. Dr. Francis TYERS

Date: January 2020

Pages: 121

In rule-based machine translation systems (RBMT), transfer rules perform transformation of source language structure into its equivalent target language structure. The grammatical, syntactic, and systematic differences between two languages, have led to the creation of these rules. The rules are applied deterministically to the input left-to-right, according to longest match. In this thesis we describe experiments applied using a two of machine learning methods (maximum entropy and support vector machine) for learning a model to distinguish between ambiguous selection of structural transfer rules in a rule-based machine translation (MT) system. Herein, the transfer rules function by matching a source language pattern of lexical items and applying a sequence of actions. There can, however, be more than one potential sequence of actions for each source language pattern. Our model consists of a set of classifiers for either maximum entropy (or logistic regression) or a support vector machine, one trained for each source language pattern, which select the highest probability sequence of rules for a given sequence of patterns. We perform experiments on the Kazakh–Turkish language pair — a low-resource pair of morphologically-rich languages — and compare our model to two reference MT systems, a rule-based system where transfer rules are applied in a left-to right longest match manner and to a state-of-the-art system based on the neural encoder–decoder architecture. Our system out forms both of these reference systems in three widely used metrics for machine translation evaluation. We also found that the maximum entropy acquired the best achievement than support vector machine.

Keywords: Machine translation, Weighting, Structural transfer rules, Ambiguous rules.

TABLE OF CONTENTS

LIST OF TABLES

<u>Pages</u>

LIST OF FIGURES

Pages

LIST OF ABBREVIATIONS

BLEU	:	Bilingual Evaluation Understudy
BP	:	Brevity Penalty
CTE	:	Caterpillar Technical English
DMT	:	Direct Machine Translation
EBMT	:	Example Based Machine Translation
CG	:	Constraint Grammar
IR	:	Intermediate Representation
MBT	:	Models With Unknown Words and Extra Feature
MNBNT	:	Models Without Unknown Words and Extra Feature
MNBT	:	Models Without Unknown Words and With Extra Feature
MT	:	Machine Translation
NER	:	Named Entity Recognition
NMT	:	Neural Machine Translation
NuSVC	:	Nu-Support Vector Classification
OOV	:	Out-Of-Vocabulary
OPUS	:	Open Parallel Corpus
OVA	:	One-Versus-All
OVO	:	One-Versus-One
OVR	:	One Versus Rest

PIET	:	Position Independent Error Rate
RBF	:	Radial Basis Function
RBMT	:	Rule Based Machine Translation
SL	:	Source Language
SMT	:	Statistical Machine Translation
SRM	:	Structural Risk Minimization
SVC	:	Support Vector Classifier
SVM	:	Support Vector Machine
TLM	:	Target Language Model
TL	:	Target Language
WER	:	Word Error Rate
LinearSVC	:	Linear Support Vector Classifier

LIST OF SYMBOLS

s	:	A sentence in the source language
t	:	A sentence in a target language
$P\ \mathbf{(}f\ /e\mathbf{)}$	:	Function that returns the conditional distribution of target sentence
r	:	Reference translation
t	:	Generated text
$d\ \mathbf{(}t, r\mathbf{)}$	:	The edit distance performs the minimum number of substitutions, deletions
hyp_k	:	The test set
ref_k, r	:	The ref set
$\|c\|$	:	The length of candidate of translation
$\|r\|$	:	The length of reference translation
$p\mathbf{(}f \mid e\mathbf{)}$	:	Function that returns the conditional distribution of target sentence
Wn	:	The weighting factor
c	:	A context in the source language made up of a sequence of patterns
Z	:	The normalizing constant
$p_s\mathbf{(}t \mid c\mathbf{)}$	:	The probability of TL word *t* being the translation of SL word *s* in context *c*
$\sum$	:	Set of input symbols
λ_S	:	A feature weight
$h^s(\mathrm{t} \mid \mathrm{c})$	:	A feature which selects the translation *t* of word *s* in context *c*
T	:	Sequence of words in TL
q_0	:	The initial state
R	:	Set of rules
x	:	Training data
y	:	The target or class
b	:	Bias
M	:	A number of classes
C	:	A parameter that controls how model fits input data
fi	:	The classifier in case SVM
F	:	A hyperplan can separate the classes

T	:	Sequence of words in TL
R	:	Set of rules
ɤ	:	Manages how close observations have to be to contribute to the classification
σ	:	Intercept constant in the sigmoid kernel
W	:	Set of words
R^n	:	A space of training data
w	:	Weight

1. INTRODUCTION

Language is the expression of ideas, thoughts, and feelings by means of speech-sounds and vocal signs combined with words. Words are integrated into sentences that make the communication easy among people. However, these speech-sounds and vocal signs forming the language is used for different forms in different geographies all around the world. Approximation of the number of languages in the world differs between 5,000 and 7,000 [1]. This number is based on a partly arbitrary distinction between languages and dialects. In order for people to be able to understand each other, there is need to be a human or a tool to translate the different language. Though human translators assist people to translate texts or speech, finding human translators is not possible all the time and it is very costly. Some of the natural language processing used around the world is as follows: Machine translation, text understanding, and generation, database natural language interface, computer assisted teaching, helping with text preparation.

This scientific study searches the nature of languages and aims to strengthen the activation and human-machine interaction. Machine translation (MT) is consider a subdomain of computational linguistics, which requires operating a computer software for translating text from first source language (SL) into second target language (TL) through a computational model of translation via an intermediate representation (IR). These translation models may be differentiated based on both their knowledge source and IR. Rule-based, corpus-based, and hybrid approaches may be based on types of knowledge used in their development [2], [3].

The predominant approach to MT is corpus-based. This includes neural machine translation (NMT), where the IR is a vector representation of the sentence, and the classic ”phrase-based” statistical machine translation, where the IR is typically the correspondence between fixed length sequences of words. These approaches operate a huge aggregations of parallel text (or bitexts) as the origin of knowledge. This parallel text is located beside its translation to learn a statistical model of translation. Creating these texts can be time-consuming and labor-intensive. However, if parallel text exists for a given a language pair in the arrangement as a tens of millions of words [4], such a system requires computational power but minimal direct human effort. In addition, by developing statistical or neural translation systems, we will ignore secondary language development, and barriers will be created between language groups. Digitalization should break these barriers, but current neural methods widen the gap between small groups of dominant languages and secondary ones.

An effective solution is to use hybrid methods that are based on combining the favorite features of two or more MT mechanisms [5]. These mechanisms either use statistical MT rules (SMT) and methods, or include augmentation of standard rule-based MT (RBMT) by using statistical knowledge [3] to diminish the aforementioned constraints. Ehara [6] stated that the combination of rule-based and statistical methods had a positive effect on translation accuracy. We followed a similar path, which involves improving RBMT systems by using statistical MT (SMT) in our approach to the research problem.

On the other hand, the RBMT system [7] can be developed relatively quickly and do not require parallel corpus; thus, RBMT system is a suitable option through constructing MT systems to perform translation between under-resourced language pairs (e.g., Breton– French, Icelandic–English, and Kazakh–Tatar) for which large parallel corpora are not readily available. Apertium is one such system that has been widely used to create systems for under-resourced languages. This system employs a pipeline architecture in which first, the text is morphologically analyzed and disambiguated; second, each lexeme is looked up in a bilingual dictionary; third, rules to select the appropriate lexeme in context are applied; fourth, transfer rules are applied to address changes in word order agreement changes; and, finally, text is morphologically generated.

In RBMT system, the translation process is based on using linguistic resources, as an example computational morphological descriptions or grammar, bilingual dictionaries, and rules for disambiguation and structural transfer.

Basically, the rules are implemented decisively to the input left-to-right longest match. When two rules are of the same length and match the same sequence, the first rule in the file is chosen as active rule. Similarly, if two shortest rule and one longest rule have matched the same sequence of tokens, the longest rule will be chosen. This case considers a disambiguation problem of structural transfer rule, because the left-right longest match rule has to be chosen in any case, and the longest rule cannot always be the result in a better translation. More details on RBMT system has given in section 1.3.1.

In addition, in RBMT system, we did not obtain an accurate translation when transfer rules applied to the input left-right-longest match. To generate a satisfactory translation, it was necessary to change the way that transfer rules were applied in the system.

Moreover, there are numerous reasons behind creating such a state-of-art MT system to implement the structural transfer rules to obtain an adequate translation in case of Kazakh– Turkish pair. First, one as a main reason is that no studies in this area have yet been conducted. Just in the same way, though Kazakh and Turkish belong to the Turkic group, they are differing enough that native speakers cannot understand the other language human or machine assistance, but Kazakh and Turkish share quite similar grammatical structure and words in which the RBMT system is feasible with some level of linguistic knowledge. In contrast, there is some apparent grammatical differences between the two languages, such as in Kazakh auxiliary verbs indicate verb tenses, however, Turkish do not have any auxiliary verbs. As a consequence, that the grammatical differences will be the main cause of having several translations of most Kazakh's verbs in Turkish, for instance the sentence *Мен* ***келе жатырмын*** 'I am coming', could be translated into Turkish either as Ben ***geliyorum*** 'I am coming' or Ben ***geleceğim***'I will come', or Ben ***gelirim***, 'I come'. During our work on this subject, I got ambition to create MT between all Turkic group in the coming days, especially for languages that do not have any translator so far, for instance, my mother tongue 'Turkmen'.

In this thesis, we describe an extension to this system where we replace the left-to-right longest match algorithm with a search of possible rule combinations. Accordingly, of this context, we suggest a new unsupervised learning approach inside of shallow-transfer MT rules have been learned automatically from monolingual corpora by using an unsupervised maximum entropy approach. Thus, the annotated development corpus is not essential for calculation. The training procedure is based on obtaining translations for whole feasible combinations into the TL by using the rest of the modules in the MT pipeline, and then acquiring normalized probabilities for all translations from a language model to switch fractional counts in the supervised learning method.

In this case, the performance of the target-language model can be surpassed by using only source-language information. Identically, we have also applied another learning algorithm to address the problem of applying transfer rules. The reason of applying another learning method is to examine how good our result is by implementing an unsupervised maximum entropy approach. As another learning algorithm, we used support vector machine approach, in which a multiclass classification algorithm such as One-versus-one and One-versus-rest are more fitted for problem of this work. Therefore, we are able to compute the accomplishment of the module and how the system acted by carrying out these unsupervised and supervised algorithms. The conflict between transfer rules is resolved by selecting the most suitable ones corresponding to a worldwide minimization function, rather than proceeding in a pairwise greedy manner.

The rest of the thesis is as follows: The first part of chapter 1 provides a brief review of previous research on the RBMT system, the SMT, and hybrid approach of integrating RBMT and SMT systems. The second part of chapter 1 presents an overview about MT and its approaches. Chapter 2 presents a preliminary evaluation tools used to evaluate the system and describe the characteristics of data used during the whole the thesis. Chapter 3 describes the system Kazakh-Turkish RBMT system. Chapter 4 describes the system and tools used to build up weighted system by using maximum entropy. Chapter 5 provides explanation of the weighted system by using support vector machine. Chapter 6 provides concluding remarks.

1.1 PREVIOUS AND RELATED WORKS

Until the first half of the nineties the translation between different languages realized by the humans. The serious research on the MT began at the middle of in the nineties with the invention electronic computer [8]. The growth of electronic computers and programming allows researches in the natural language processing and its sub-field MT area. Furthermore, since developed systems are achieved for a small collection of language pairs, there are many language pairs that do not have MT system yet.

Natural language processing involves morphological analysis, definition, correction, sentence structure research, and translation operation. MT is the broadest processor and researcher topic for natural language processing. Nowadays, although in the MT field there are systems that can produce successful results in specific domains, for general use they have not been developed yet. For instance, METEO system was the first successful MT system that was developed and become functional in 1976. They developed a MT system specifically for daily translating of the weather reports from English to French. Though, the grammar and vocabulary for the language in this report was very limited, the METEO system was successfully used. Also the system of English to Japanese for translating assert topics was 98% and have been actively used in scientific research [9].

1.1.1 Prior Researches of Machine Translation System Based on Rule-based

MT systems have relied on handcrafted rules. In which transfer rules determine how a (syntactic or semantic) structure in a particular language maps to the corresponding structure in another different language. In particular, handcrafted rules used in translation systems their purposes had limited function in specific areas of science Systran system [10] is an example of this approach, which is a new sketch based on the conventional Systran MT system. This system was used in development for English-Hungarian, English-Polish, English-Arabic, French-Arabic, Hungarian-French and Polish-French language pairs. Furthermore, approaches for building rule-based systems have been studied, such as Open Logos [11], [12], Grammatical Framework [13], and Nooj [14]. All these approaches have been using grammatical knowledge and lexicon of languages in translation.

Studies on RBMT are ongoing. For instance, the rule-based English-Koreen MTS [15] can outperform the NMT system in improving translation quality by analysis and processing comma for long sentences. Mukta et al. [16] presents an ideal method to obtain the best translation from English to Bangla by using RBMT. Knowledge-based technique has been used with a set of data to classify each English sentence to a specific group using features and organize these in a pattern. This method exhibited higher accuracy than Google Translate. Authors in the article Son et al. [17] presents a rule-based Vietnamese-Thai MT system that uses a syllable matching algorithm together with named entity recognition (NER) for segmenting the output sentences into sequence words by punctuation marks. Vietnamese-Thai transcription rules have been used to transcribe unknown words and recognized named entity words. Their translation also was much better than the results obtained through website Google Translate.

Hurskainen et al. [18] introduced an English-Finnish rule based MT. Lexicon and grammar of SL and TL were used in the creation of the system. Source sentences for English were adapted through subsequent phases into the TL. They never used statistical choices in the MT. To reduce needless rule writing, defaults are employed where feasible. The researchers concluded that the more grammatical sentences are, the better the result will be.

Within the scope Apertium project, ongoing work is building MT i.e., Components underlying the MT such as morphological transducers systems for translating between two Turkic languages, either between a Turkic language and Russian or between a Turkic language and English, and also between several other language pairs. The MT systems being developed for Turkic languages are for Turkish-Kyrgyz, Azeri-Turkish, Tatar-Bashkir, some of the released MT systems are for Kazakh-Turkish [19], Kazakh-Tatar [20], Tatar-Bashkir [21], Crimean Tatar-Turkish and English-Kazakh [22].

Various other MT systems have been documented that for Turkish and other Turkic languages, including Turkish–Crimean Tatar [23], Turkish–Azerbaijani [24], Turkish–Tatar [25], and Turkish–Turkmen [26]. Tantuğ et al. [26] used a Hybrid Model that applies an association of rule-based translation and statistical translation methods. In this system, when a translation is developed between any Turkic languages and Turkish, a support base structure is created which helps using Turkish without any additional change in the system [27]. The infrastructure of the hybrid system is also used in the translation system between Uyghur and Turkish [28]. For the systems used in translating into or from Kazakh (besides those already mentioned in this paper), a bidirectional Kazakh-English MTS [29] uses a link grammar method and a statistical approach. To the best of these MT systems of our information, none of these MT systems have been released to the public.

Altenbek et al. [30] proposed a segmentation system for inflectional affixes of Kazakh. Washington et al. [31] presented work on morphological analysis for three Kipchak languages, namely, Kazakh, Tatar, and Kumyk.

Moreover, MT systems between different languages than mentioned previously in this paper and they are within Apertium or using Apertium as a source. For instance, Centelles et al. [32] created a Chinese-to-Spanish rule-based MT system that combines manual and statistical techniques. Notably, the researchers used human knowledge in addition to statistical knowledge to provide and create monolingual and bilingual dictionaries and defined grammatical transfer rules as the main component of the MT system. Text from different domains have been used for evaluation, which showed that the RBMT has high coverage in various domains. Toral et al. [33] presented an Italian-Catalan rule-based MT system which built automatically by associating the linguistic data of the existing quad Spanish–Catalan and Spanish–Italian.

After data post-processing, the researches were able to address inconsistencies in the automatically derived dictionaries. At the same time, the researches add frequently used words which were absent in accordance with a corpus analysis. Moreover, TER [34] and GTM [35] were used to evaluate the system, which outperformed Google Translate by 10 absolute points.

1.1.2 Prior Researches on Statistical Machine Translation

Statistical machine translation (SMT) emerged at the beginning of the 1990s and has registered good achievements in MT performance. Examples of SMT systems include Candide [36] and SBTG [37].

Therefore, some recent studies have been accomplished between Turkic and other languages. This article [38] is considered one of the recent studies to apply this approach, in which the authors present English-Turkish MT and exhibit the result of using N-best list on the MTs. When SMT systems generate more than one translation for input sentences in the SL, several different translations are held in an ordered list known as N-best list. Such a list determines the effect of reordering possible candidate translations on MT success, in which the top-ranked candidate will likely be selected as best probable translations. The first probable translation may not be the most appropriate and meaningful translation. Thus, the authors reranked the N-best list to select neglected translations that are presumably better. The intention was to discover a reranking process on N-best lists that would produce a better performance on the English-Turkish MT. Google

Translate Research API1 was used as the SMT system to obtain the N-best lists, thereby enabling the authors to achieve practical performance.

1.1.3 Integration of Rule-based Machine Translation into Phrase-based Statistical Machine Translation

All the RBMT and the SMT systems have strengths and weaknesses in providing the best translation. For this reason, the research community has switched its concentrate toward integrating rule-based and statistical methods by combining of their outputs of MT systems, which are known as hybrid systems. The translation qualification of MT has been improved dramatically by using hybrid systems. Interdependencies between two systems work with one of the systems in process conducting the translation procedure and the other ones reinforce overall translation quality. In one approach, some studies have been conducted on constructing systems which the statistical constituent is responsible for the translation and the accomplice system supplies completing information. For example, Elsele et al. [39] and Chen et al. [40] introduced lexical information from a rule-based translator into an SMT system in the form of new phrase pairs for the translation table. In both cases, the results have been positive on out-of-domain tests. The other approach, in which the translation would be led by RBMT system and complementary information will be provided by the SMT system, has been less explored. Habash et al. [41] improved the dictionary of an RBMT system with phrases from an SMT system. Globally, the results improved the particular systems while the hybrid system applying translation into languages with wealthy morphology than the source.

Park et al. [42] proposed a hybridization approach based on classification method by integrating an SMT system and an RBMT system. This work addressed a labeling issue by creating a training dataset. Labels used in training datasets were assigned to the best translation result. The labels were produced by using BLEU metric because BLEU favors SMT translations over RBMT, because BLEU and SMT similarly inherently opt fluency over accuracy. Such a case considers a certain risk because the quality of RBMT has been undervalued. This issue has been resolved this issue by using cutoff method. Thus, metric evaluation scores are given for two translations (one from SMT and the other from RBMT), the highest evaluation score for translation will labeled as

SMT system production. That was not important if the process was located first or not. This situation occurred for entirety translations have an evaluation score larger than a certain threshold (cutoff point). With regard of this issue, an uncertainty or error in the classification has been solved, and through the feature groups, the effects on learning the classifier for hybridization was controlled. Using the aforementioned cutoff method, the researchers achieved improvement in translation accuracy and overall quality.

Ahsan et al. [43] coupled an English-Hindi RBMT system and a standard phrase-based SMT system in the analysis, transfer, and generation levels in the RBMT pipeline. In the analysis stage, the source analyzer conducted linguistic analysis. In the second stage, local and long-distance reordering have achieved by the Transfer Grammar module, in which the source sentence converted into chunks in a dependency structure. In the last stage, the task of lexical transfer of source sentences processed by generation component maps them into target strings, thereby helping authors find out the typologically divergent English-Hindi language pair. The effects of the source transformations have been seen by authors at each level on the performance of the coupled MT system. Researchers used extremely small datasets in their experiments, and their evaluation results show significant improvements in terms of BLEU (RBMT 7.14 and 0.87 and SMT baselines respectively). In Another study on hybrid systems, Banik et al. [44] proposed a new approach of serial coupling by building a hybrid model of an English-Hindi pair, which enabled them to create an effective hybrid system that exploit the advantages of SMT and RBMT. The system is able to generate catalogues from English to Hindi, which is a difficult and challenging task due to the nature of the domain.

The structure of the coupling system consists of three parts. The first part was used to obtain good lexical selection and robustness. In the second part, syntax was improved. The final part was used to integrate other modules along with the best phrase reordering. The proposed method exhibited n improvement in BLEU score on an English-Hindi product domain dataset.

1.2 MACHINE TRANSLATION

In recent years, researchers have been working on MT systems for many of the world’s languages. At first, people thought a message that was written in different language as having basically been written in their own language, in an enciphering style. Thus, the translation procedure was a process of deciphering the enciphering form of messages. But, after that they recognized, the translation procedure is much more complicated than just decrypting [45].

MT is a process of translating written text from one natural language to another with assistance of computers. MT makes communication amid people which speak different languages much easier than the past years. Therefore, today, all studies in this area aim to provide easy interaction between people.

The MT is most recent application of processing natural languages of artificial intelligence [7]. The MT between two languages called bilingual which can be either unidirectional or bidirectional, and called multilingual, when MT has formed for more than one pair of languages. MT is much substantial for multilingual communities because it will be able to translate immense amount of texts written in SL into TL in the short time period, however, this process not feasible with human translator.

The RBMT system and Empirical Based MTs are two predominant approaches of MT. Along with, the Hybrid MTs consists of the RBMT system and the Empirical Based MTs. In RBMT, the linguistic rules based on the morphological, syntactic and semantic information and this method has many types of approach like (Direct approach, transfer based approach, and Interlingua based approach). The SMT system, Example-based machine system (EBMT) and the NMT system are subdomains pf the empirical Based MTs. Using appropriate methodologies and with limited resources to develop efficient MT systems is a big challenging task, particularly for a morphologically rich language like Kazakh and Turkish. Particularly handling the difference structural between the two languages and processing the ambiguity are the two major difficulties in MT [46].

1.3 APPROACHES TO MACHINE TRANSLATION

1.3.1 Rule–Based Machine Translation

Transfer-based RBMT systems are categorized based on to the complexity of the IR used in shallow-transfer, syntactic-transfer, and semantic-transfer RBMT systems. Our work focuses on Shallow-transfer RBMT systems, which are those that accomplish a shallow-syntactic analysis of the SL, i.e., they do not achieve any syntactic parsing and do not construct a parse tree. The IR used indicates that the process is as basic as a series of lexical forms (lemma, lexical category, and morphological inflection information) of the words to be translated. Transfer rules normally divide this series into chunks (groups) of lexical forms with elements that are processed together.

Shallow-transfer RBMT systems implement process of the translation in three main levels: the SL is analyzed to generate an SL IR, the SL IR is transferred to a TL IR, then, the generation of the final translation form TL IR. Shallow-transfer RBMT systems employ basic IRs consisting of a sequence of lexical forms and do not achieve a complete syntactic analysis of the input sentences. These systems use bilingual dictionaries for lexical transfer, and shallow-transfer rules for structural transfer. The structural transfer rules apply to chunks of lexical forms in SL and outcome TL lexical forms. These rules are typically written by domain experts who are either translators, linguists, or language engineers. Depending on the availability of existing resources and the expertise of the developers, these systems can be developed anywhere during a period of several weeks, several months or longer. According to the authors experience of the authors that building an RBMT system rarely takes as long as is suggested in the literature. Most estimates of “several years” include the time taken to build all of the existing resources, which is as if the time taken to produce the parallel text were included in the time taken to build a corpus-based system.

The transfer rules task involves transferring to SL structure into its equivalent TL structure. The grammatical, syntactic, and systematic differences between two languages, such as Kazakh and Turkish, have led to the creation of these rules. The transfer rules perform a transformation of the lexical form which is the output of morphological analysis by applying some rules for the SL to produce a surface representation of the TL. The transformation process consists of a pattern, which matches a fixed length sequence of SL lexical units and an action which performs operations on this sequence, then rules are applied deterministically to the input left-to-right longest match. If two rules are of the same length and match the same sequence the first rule in the file is selected. Left-to-right longest match is a good heuristic, it can be the case that there can be more than one possible set of actions can be applied to a matched sequence and a different combination of rules can result in an improved translation.

The transfer rules task involves transferring to SL structure into its equivalent TL structure. The grammatical, syntactic, and systematic differences between two languages, such as Kazakh and Turkish, have led to the creation of these rules. The transfer rules perform a transformation of the lexical form which is the output of morphological analysis by applying some rules for the SL to produce a surface representation of the TL. The transformation process consists of a pattern, which matches a fixed length sequence of SL lexical units and an action which performs operations on this sequence, then rules are applied deterministically to the input left-to-right longest match. If two rules are of the same length and match the same sequence the first rule in the file is selected. Left-to-right longest match is a good heuristic, it can be the case that there can be more than one possible set of actions can be applied to a matched sequence and a different combination of rules can result in an improved translation.

In essence, direct-based, transfer-based, and interlingua are three subcategories of the RBMT system. Currently, the most commonly used methods and their differences is depicted in Figure. 1.1.

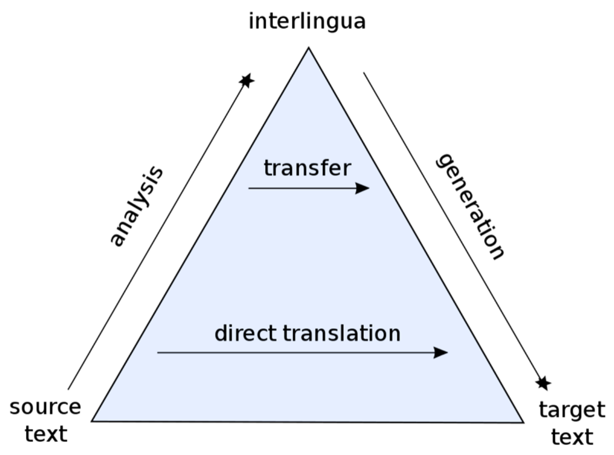


Figure 1.1: Vauquois triangle

1.3.2 Direct Machine Translation

In this approach the translation is direct from source (SL) to target (TL) with some little syntactic or semantic analysis. The translation realized on word level. Words are translated into the TL without passing through an IR. After carrying out the morphological analysis of the SL, the SL is translated directly to the TL without performing any further processes [47]. Direct machine translation (DMT) system can be either uni-directional or bidirectional. Developing the DMT system is straightforward and this property consider as a good aspect of this system. Whereas an ambiguous problem in word transfer stage have not been solved yet and this behavior is counted as drawback of the DMT. For instance, in Kazakh the word астана in Turkish can be translated either as *başkent*, 'city name' or *merkez*, 'center'. Therefore, when the word translated into different sense, the sentence will lose its appropriate meaning. Anusaarka system is a DMT, which has been developed in Indian Institute of information Technology, Hyderabad. The Anusaarka system is built as direct approach and cover and covers all major Indian languages [1].

1.3.3 Interlingual Machine Translation Approach

The main reason behind building Interlingual MT system was the failure of the direct translation system. This excuse led to the develop a sufficient linguistic model for translation. In the interlingual MT system, the text which is an example of SL firstly transformed into aniInterlingual language “language neutral” representation which is liberated from every language and then, a TL generated from interlingual language. In addition, the interlingua based model composes a parse tree from the SL. The interlingua based model goes a further step and transforms the SL parse tree into a standard language-independent format, known as interlingua [48].

The fundamental idea of an interlingual method is the translated text reflects the meaning of the translated text in all world languages. Thus, the target text from this interlingual representation can be generated into any language that this interlingual MT system will be able to generate it. Because, the SL has to be analyzed accurately, the interlingual MT system is supported by the knowledge base. Therefore, every detail of the source text such as syntactic information and likewise the purpose which is illustrated in the interlingua representation should be captured. For example, the word in a SL (Turkish) *ekmek*, ‘bread‘’ translated into interlingua (intermediate language) as ‘olay‘’ and this translate into a TL (Spanish) as pan, ‘bread‘.

Nevertheless, KANT system is the interlingual MT that made processes at the commercial level. the interlingual MT is designed to translate Caterpillar Technical English (CTE) into other languages [47].

1.3.4 Transfer-Based Machine Translation

The transfer-based approach works like interlingual MTs that built the translation from an IR which replicates the meaning of the original sentence, the source text transferred to a representation and this representation is then generated for the TL using bilingual dictionaries and grammatical rules. Because of the structural differences between the SL and the TL, a transfer system could be accomplished in three steps: i) Analysis: In this stage, the SL is parsed, the component and sentence structure of the sentence are identified based on the linguistic information such as morphology, part-of speech, syntax, semantics etc. ii) Transfer: The transformation is employed on the SL parse three for the structure adaption in the TL in other words the syntactic/semantic of SL is moved into the syntactic/semantic structure of the TL. iii) Generation: In this level words are translated and represents the gender, number, tense etc. in the TL.

In the analysis step, the SL parser is applied to get the more accurate meaning from the source. In the transfer stage, the transfer rules are applied to this analysis to convert it to the TL-oriented representation and final stage of this procedure is the generation of the target text. When the languages are close to each other the performance will be high [45]. There are several systems use transfer based machine translation such as GETA MTs [49] and, SUSY MTs [50].

1.3.5 Hybrid Machine Translation

Hybrid MTs uses benefits of two techniques, The RBMT system and the SMT system, in which the Hybrid MTs has proved to have a good efficiency in the field of MT systems. The Hybrid MTs takes advantages of applying rules and using learning approaches. There are different ways the hybrid MTs can be used for achieving its high performance to get a convenient translation. The hybrid MTs can be used in a number of various manner. In some states, translations are accomplished in the first step applying a rule-based method and that will be chased adapting or correcting the output using statistical information. In the other states, rules are applied to pre-process and post-process both input and output data of a statistical-based translation system. The second approach is preferred on the previous one and has extensive power, flexibility, and control in translation [51]. Nowadays, several research has been working by combining both of both approaches (RBMT, and SMT). Also, assorted governmental and private based MT sectors have involved to use this approach to advance translation quality from SL to TL. The drawback of this system that it is costly. The Hybrid MTs is integrating more than one MT paradigm, this is to make the Hybrid MTs more effective and pulls on attention. An example of hybridization MTs is a METIS-II MTs which is situated on EBMT framework; METIS-II MTs use a bilingual dictionary (similar to that identified in most RBMT systems) and a monolingual corpus in the TL [52]. The Oepen MTs is an instance of hybridization MTs over the rule-based paradigm. The Oepen MTs integrates the statistical methods inside an RBMT system to choose the favorite translation from a series of contest hypotheses (translations) and generated using the rule-based methods.

1.3.6 Statistical Machine Translation Approach

SMT [53] is used frequently as a type of corpus-based MT. The translation process in SMT performs on the basis of the information automatically learned from previously translated texts.

Building an SMT requires minimal human labor when adequate monolingual and parallel corpora are usable and therefore this system is always in demand. In this work, which is based on the statistical models extracted from parallel aligned bilingual text corpora, the assumption is that every word in the target language is a translation of the source language words with a certain degree of probability [54]- [56]. Thus, the words with high probability will be selected. In information such as translation rules, the dictionary and context information for each word can be extracted from an adequately huge corpus. Nevertheless, parallel corpora are not consistently available and may not even exist for more (under-resourced) language pairs. The system is ready to perform actual translation after it is trained with large corpora. Two types of essential alignments are conducted. One is a sentence alignment for the bilingual texts at the sentence level, and this consider a word alignment for each source word in the target language. The system with the aligned corpus learns how to achieve translation process of words, phrases and grammar rules. The repetition of every two words that appear together can be extracted from this corpus and the rules can be applied to the existing text. Then, the system performance is improved by adding the translation results to the training corpus. Moses [57] is an example of an Oepen MT system.

Supervised or unsupervised algorithms are implemented to create a statistical table from the corpora and the current process named either learning or training process [58]. The content of tables includes statistical information like the correlation between languages and characteristics of well-formed sentences. The three common types of this model such as: Statistical word-based translation model; Statistical phrase-based translation model; Statistical syntax-based model. In general, the SMT system rely on the following models that are a language model, a translation model, and a decoding algorithm. The translation model guarantees the target hypothesis of MTs corresponding to the source sentence. Additionally, the grammatically correct source sentence can be insuring by the language model. This approach can be constructed for any language pair and with minimal human effort. The SMT model never applies any linguistics analysis on input text and it works on the available parallel corpora. The SMTs have the ability to appreciate indirect knowledge incorporated in a co-occurrence statistic.

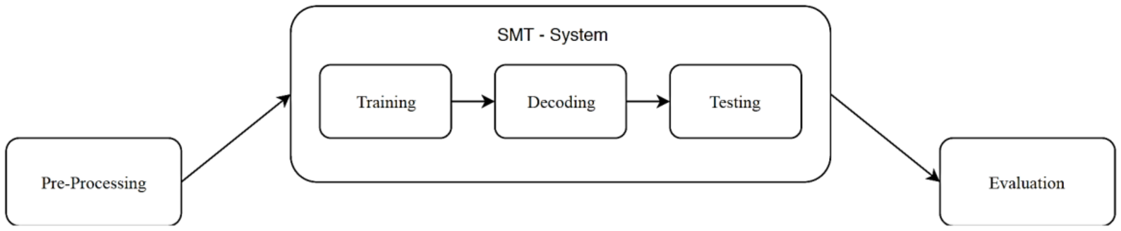


Figure 1.2: SMT System

The first model of SMT, was based on Bayes Theorem that says, every sentence in one language is a probable translation of each source sentence and the best convenient one, is the translation which is assigned the highest probability by the system. The drawbacks SMTs are: a SMTs have difficulties to manage the circumstance which require linguistic knowledge as morphology, syntactic functions and ordering of words. Guessing the probability of a translation, and efficiently finding the sentence with the highest probability. Figure 1.2 depict the practical stream diagram of an SMT system.

Although a large parallel corpus is existent, SMT systems still have some limitations as a result of (i) the data scarcity problem that causes difficulty in collecting enough phrase pairs that cover all the inflected word forms in highly inflected languages, and (ii) the domain problem caused when the training parallel corpus be part of a domain different from that of the texts to be translated. Thus, efforts to find ways to avoid the problems of statistical systems are needed [59].

1.3.7 Example-based Machine Translation Approach

The EBMT approach is depending on recalling and discovering analogous samples of bilingual corpus with parallel text. In 1981 Makoto Nagao proposed the concept of "translation by analogy" [37]. In this approach, the EBMT system is supplied a group of sentences of a SL (from that one is translation) and equivalent translations of every sentence in the TL along point to point mapping. These examples have been used to translate analogous kinds of sentences of a SL to a TL. The basic hypothesis is that, if a previously translated sentence occurs again, the same translation is likely to be correct again [60].

The EBMT system has four species: example base and management, example acquisition, example application, and synthesis. The idea of the EBMT system is translation accomplished with analogy. The assumption of translation by similarity is encrypted to example-based MTs over the example translations that are used to train such a system. The EBMT system is an appealing way of translation because it prevents a requirement for manually constructed rules. The one drawbacks of the EBMT system are the analysis and generation modules are needed to produce the dependency trees needed for the examples database and for analyzing the sentence. Other poor situation with EBMT system is computational efficiency, especially for huge databases, although parallel computation techniques can be applied.

1.3.8 Neural Machine Translation

The NMT system is considered a current approach for corpus-based machine translation which was first released by [61], [62], and sequence-to-sequence models is used in its development [61], [63]. A single, large neural network which reads a sentence and yield appropriate translation is created and trained by the NMT system. In fact, the majority of the prospective NMT models appertain to a classification of encoder–decoders [62], [63], with an encoder and a decoder for each language. The source sentence is encoded into a fixed-length vector by encoder neural network. Similarly, a decoder neural network will decode a translation of the encoded vector. Notice that, the encoder-decoder neural network system, that refer to the encode and the decode of a language pair, is collectively trained to boost the probability of an appropriate translation via given a source sentence.

Moreover, some recent researches have applied neural network to directly learn the conditional distribution $\boldsymbol{p}\ (\boldsymbol{t}|\boldsymbol{s})$ of a target sentence (translation) t given a source sentence s from a bilingual, parallel corpus [61, 62]. Kalchbrenner et al. [61] used a convolutional n-gram model to obtain a fixed-length vector of a input source sentence which is decoded with an inverse convolutional n-gram model added with an RNN. Similarly, Sutskever et al. [62] purposed an approach associating and RNN with LSTM components was applied to encode a source sentence and beginning from the last hidden state, to decode a target sentence. As well as, Cho et al. [63] had utilize RNN to encode and decode a pair of source and target phrases. Basically, all the researches mentioned above based on encoder–decoder architecture see Figure 1.3. As depicted in Figure 1.3, the encoder handles a variable-length source sentences and create a fixed-length vector representation which is denoted as z and the decoder produce a variable-length sequence target sentence.

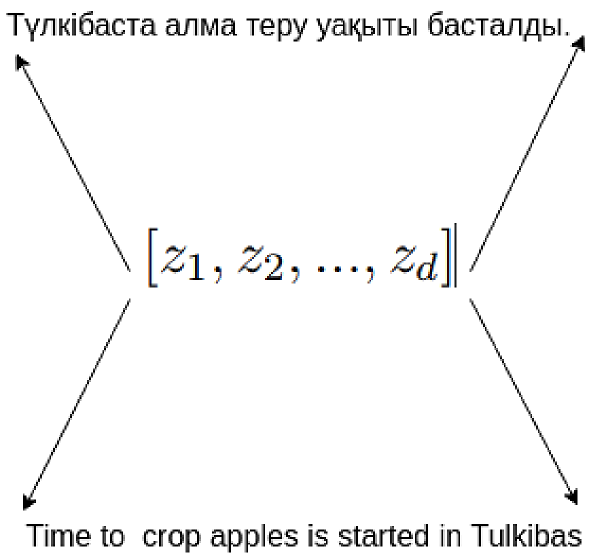

Түлкібаста алма теру уақыты басталды.
$[z_1, z_2, \ldots, z_d]$
Time to crop apples is started in Tulkibas

Figure 1.3: The encoder and decoder structure

The NMT approach encounters difficulty handling long sentences because the fixed-length vector representation does not contain sufficient competency to encode all the necessary information of lengthy sentences with complicated structure and intent. Cho et al. [63] justify that absolutely accomplishment of the encoder–decoder neural network decline speedily as the length of an input sentence increases. Furthermore, the performance of NMT system decreases rapidly as the number of unknown of words increases. This issue presents a challenge in increasing the magnitude of vocabularies employed by NMT systems in the future. Another limitation of the NMT system is its incompatibility with the specific uses of certain organizations, thereby causing difficulties for them to refine and improve the system according to their needs.

2. EVALUATION SETTING

This chapter states exhaustive illustration about data and evaluation settings which have been used in the rest of the thesis. The various automatic methods are projected to evaluate the MT quality by comparing hypothesis translations with reference translations. In the past, human justices used to examine the quality of machine translation systems. The humans considered many prospects through their evaluation of MT such as adequacy, fidelity, and fluency of the translation. The human's justice for evaluations of MT are extensive but pretty expensive and they would take weeks or months to accomplish. This contemplate big issue because developers of MTs require to manage the effect of daily changes to their systems in order to decrease poor concepts from good concepts. The objective of Human evaluations of MT has boost many scientists to develop reliable methods for estimating such measures automatically.

Nowadays, people prefer to use automatic evaluation methods to measure the MT quality through comparing hypothesis translations with reference translations. These methods automatic are repeatable, and be performed over a large test data, these methods in demand more than human evaluated MT. Examples of such methods are word error rate (WER), position-independent word error rate (PER) [64], generation string accuracy [65], multireference word error rate [66], Bilingual Evaluation Understudy (BLEU) score [67], NIST score [68]. The BLEU, WER, and PER are by far the most widely used metrics. Unquestionably, all the automatic metrics above attempt to inexact human assessment and often perform a marvelous point of correlation to human subjective evaluation of fluency and adequacy [67], [68].

A common dilemma of computing performance of MTs lies in the fact that the translated word sequence can have a different length from the reference word sequence (supposedly the correct one) [69]. The WER metric is assumed from the Levenshtein distance, performing operation at the word level. Likewise, the PER metric is calculated on a sentence-by-sentence level. The main difference between two metrics (WER and PER) is that the PER does not punish the wrong order in the translation. Accordingly, during the whole work, we realize the evaluation of all MTs by comparing the produced test data by our MTs with postedited versions of the same test data.

The entire evaluation mission is an extrinsic evaluation, herein, the evaluation metrics test the performance of the MT systems regularly as regards eventual translation quality in an original system. The whole evaluation methods which is explained in this thesis be as appropriate to less-resourced and marginalized languages, as to dominant language. The chapter starts with a brief explanation of the corpora that we use for training and testing, also the evaluation will subsequently be described. This will be followed with a description of the automatic evaluation metrics which are used in evaluation.

2.1 CORPORA

- **Wikipedia dumps** the wiki dumps files are free available corpora. The Wiki dumps files very convenient for quickly obtaining a required data, particularly the pages' article file which produce data (text corpus) for training unsupervised part-of-speech taggers, n-gram language models, etc.

To create the training corpora, we provide an SL corpus for training and a TL corpus for scoring. We used two freely available corpora: a dump of articles from Kazakh Wikipedia with size 320 MB, 643.4 MB, and a single dump of articles from Turkish Wikipedia with size 440.6 MB were used for scoring TL corpus. For good performance, we need to train the system with huge data. Because, the appropriate size of data for single corpus of SL was not available, we trained the system with two different sizes of data.

The training phase consists of setting up a co-occurrence model of SL lemmas (with the equivalent scores) for each translation sense managed by the MT system. As our application takes a sentence as input, we must break the corpus into sentences. To perform this task, we applied a rule-based sentence boundary detection tool called a **pragmatic segmenter**. This tool also removes unnecessary characters like ('<' , '>' , '$', '#', '@', '*', '{', '}', '+', '^', '/') which would crash the system. Identically, these characters have appear escaped in a partum input stream unless they are part of a lexical unit, chunk or super blank.

Then, we have randomly selected 1,000 sentences pairs for testing the performance of the system. Table 4.1 introduces the statistics of the test corpora, specifically the number of sentences used for testing, and the number of tokens in the source and target sentences. The number of ambiguous tokens indicates the amount of singular tokens with more than one likely translation. Then, we calculated the mean number of translations for each ambiguous word by dividing the number of ambiguous tokens by the entire number of tokens.

Table 2.1: Statistics of the test corpora used by three systems (Weighted, NMT, and Moses). The columns SL and TL present the total number of tokens in the source and target languages. The columns "No.amb" and "% am-big" indicate the number of words with multiple translations and the percentage of SL words that have multiple translations, respectively.

	Lines	SL	TL	No. amb	%am-big.
Test	1000	9,158	9,249	1,619	5.65

Table 2.2: Statistics of the training parallel corpora used by three systems (Weighted, NMT, and Moses). The columns SL and TL present the total number of tokens in the source and target languages, and the column "Dev" indicates the number of sentence pairs used as development data by NMT and SMT. The columns "No.amb" indicates the number of words with multiple translations.

	Lines	SL	TL	DEV	No. amb
Train	62,893	266,55	285,6	5000	9,999

As we have compared our weighted system with neural machine translation (openNMT) [70] system, and with another statistical machine translation system Moses [71], we used a parallel corpus for Kazakh and Turkish with size of 3.2 MB which is available online through OPUS5. All details about OPUS is given in section 2.1. Table 2.2 presents statistics for the parallel corpus that we used in NMT system, SMT system, and weighted system training. After splitting the corpus into training and development sets, we selected 5,000 sentence pairs for development (dev) and left the rest for training. Approximately 266,000 tokens are used for each language in the training corpus, and 18,000, each for the development corpus. Moreover, we calculate the number of ambiguous and average of ambiguous words over the entire corpus. The number of ambiguous words indicates the amount of unique tokens with several possible translations. The column average ambiguous indicates the average number of translations every ambiguous word. This can be calculated by looking up per word in the corpus in the bilingual dictionary of the MT system and dividing the total number of translation by the number of words.

- **OPUS (the open parallel corpus)**

  OPUS is based on open source products and the corpus is also brought as an open content package. To test how well RBMT (weighted is either with maximum entropy algorithm or with support vector classifier algorithm) as compared with NMT, and other SMT, we trained all of the base-line mentioned machine translation systems using corpus has obtained from OPUS [75].

Since we have compared our system with neural and statistical machine translation systems and for that we used parallel corpus for Kazakh and Turkish with size 3.2 MB form KDE4 which is available online through OPUS.

Table 2.2 presents statistics of the parallel corpus that we use in NMT system and SMT system training. In addition, to testing the systems, we use the test dataset presented in the Table 4.1 that compared weighted and other reference systems in the next chapters. There are number of reasons for not using the parallel data from OPUS for testing. First, many repeated sentences occur in the source language corpus. Second, some of the target sentences are not the right translations for the source sentences, and they are absolutely irrelevant parallel sentences. Therefore, we used the same previous test dataset as presented in the Table 4.1 for the weighted, NMT, and Moses systems testing after training with current parallel data.

2.2 REFERENCE RESULTS

We compare our system (either with maximum entropy or support vector classifiers) with the reference (or baseline) systems as described below:

- **Linguist-chosen defaults (unweighted).** A structural transfer in an Apertium language pair contain of rules with a similar pattern's name. In this case, more than one rule can apply into same the token, and that means one sentence can have more than one translation. But a single sentence form may not have more than one translation without further processing. If there are many possible translations of a structural form, then one must be marked as the default translation.
- **Linguist-chosen random.** A structural transfer in an Apertium language pair contains rules with a similar pattern's name. And because more than one rule can apply to same token, that means we can obtain more than one translation for one sentence. And here we are just going to choose randomly one of these translations.
- **Target language model (TLM).** After using one of the structural transfer methods for getting all possible translations from our system, we used language model method on-line to score these target sentences. The translation with supreme score has chosen as best translation. This is the method used by [73]. We have used six six-gram language model to score the generated TL side. This system has given a very efficient result compared with other systems. One method of structural transfer is to use the existing MT system to generate all the possible translations for an input sentence, and then score these translations on-line on a model of the TL. The highest scoring sentence is then output. This is the method used by [73].

2.3 PERFORMANCE MEASURES

The translation quality is measured using three different metrics: the WER and PER metrics. Both metrics are based on the Levenshtein distance [74]. Metrics based on WER were selected to compare the system against systems based on similar technology and to assess the usefulness of the system in a real setting, that is, to translate for dissemination. We took a small number (9,158 tokens) of Kazakh text, which was a concatenation of several articles from Wikipedia, and translated it using the four MT systems. The output of each system was postedited independently to avoid bias in favor of one particular system. Then, we calculated WER and PER for each using the each **apertium-eval-translator** tool, and we applied the widely used BLEU metric tool, which ideally would test how much the system improves as regards an approximate measurement of the final translation quality in a real system [75]. Note that, the BLEU score is typically calculated by comparing the translation quality against a pre-translated reference translation.

We have used **apertium-eval-translator. line** for calculating WER and PER for all three systems (weighted, NMT, and SMT) as presented in Table 4.5 at Chapter 4 because the sentences were aligned, and each sentence was considered individually and takes an average of the result. We used parallel data for training the systems. Following we give very details information about the entire metrics that we used during our experiments.

2.3.1 Standard Word Error Rates (overview)

WER and PER the most widely used among a variety of automatic evaluation measures have been studied over the last years. WER and PER are measures of post-edition effort - here lower scores are better. Both metrics are based on the Levenshtein distance [74], the edit distance ***d* (*t*, *r*)** which is the minimal number of substitutions, deletions and insertions that have to be performed to convert the generated text (produce translation) ***t*** into the reference translation ***r*.**

The metrics depend on word error rate were preferred as being able to compare the system against systems rely on identical technology and to determine the usefulness of the system in a real setting, that is, to translate for dissemination. The defects of WER are that: First, it does not perform any reordering of words, though the word order of the generating sentence can be different from word order of the reference even though it is correct translation. Second, it depends fundamentally on the choice of the sample translation. PER also does not reorder the words in the sentences, but it compares the words in the two sentences. It is always less than or equal to the WER. On the other hand, shortcoming of the PER is the fact that the word order can be important in some cases. Accordingly, the best way to solve this problem is to compute both word error rates [76].

Table 2.3: The word error rate plus position independent error rate is computed for the source sentence and the two sets of translations. The source sentence **S = (*s*₁, *s*₂, ..., *s*|S|)** has one ambiguous word, секіргенді. There is one distinction between the references set ***ref*$_{k,r}$** and the test set ***hyp*$_k$** of translations, therefore, the error rate for this sentence is 16.67%.

S	*Мен*	*акуланың*	*су*	*бетіне*	*секіргенді*	*көрдім*
refk, r	{Ben}	{köpekbalığının}	{su}	{yüzüne}	{sıçrayarak}	{gördum}
hyp$_k$	{Ben}	{ köpekbalığının }	{su}	{ yüzüne }	{sıçradığını}	{ gördum }

- **Word Error Rate (WER)** The WER is calculated of the hypothesis hyp with regard to the reference ref by Eq. 2.1:

$$Wer = \frac{1}{N^*_{ref}} \sum_k^{k=1} m_r in \, d_l \, (ref_{k,r}, hup_k) \qquad (2.1)$$

Herein ***d$_l$* (*ref*$_{k,r}$, *hup*$_k$)** is the Levenshtein distance between the reference set ***ref*$_{k,r}$** and the hypothesis sentence (test set) ***hyp*$_k$**. Indeed, calculation of WER is accomplished using a dynamic programming algorithm.

- **Position independent Error Rate (PER)** The PER has been measured using the counts ***n* (*e*, *hyp*$_k$)** and ***n* (*e*, *ref*$_k$)** of a word e by using Eq. 2.3 in the hypothesis sentence ***hyp*$_k$** and the reference sentence $ref_{k,r}$ respectively with Eq. 2.2 :

$$Per = \frac{1}{N^*_{ref}} \sum_k^{k=1} m_r in \, d_{per} \, (ref_{k,r}, hyp_k) \qquad (2.2)$$

Where

$$d_{per} \, (ref_{k,r}, hyp_k) = \frac{1}{2} \left(|N_{ref_{k\,r}} - N_{hup_k}| + \sum_e |n(e, ref_{k\,r}) - n(e, hyp_k)| \right) \qquad (2.3)$$

2.3.2 Bilingual Evaluation Understudy

The BLEU [75] is a metric for evaluating a generated sentence which has machine translated from first natural language to second to a reference sentence. The better quality of translation is obtained when the machine's output is closer to human translation [75]. BlEU is one of the oldest metrics to realize high equating with human judgments of quality, and still one of the most important automated and costly metrics. BLEU based on a modified form of precision [75], [77]. For computing the precision, BLEU counts the number of candidate translation words (unigrams) which happen in each reference translation and then divides by the total number of words in the candidate translation [75]. Moreover, BLEU measures how well a machine translation coincides with multiple human translations using n-gram co-occurrence statistics. N-gram precision in BLEU is computed by Eq. 2.4 as follows:

$$P_n = \frac{\Sigma_{c\epsilon candidate}\Sigma_{n\_gram}\, count_{clip}(n-gram)}{\Sigma_{c\epsilon candidate}\Sigma_{n\_gram}\, count(n-gram)} \tag{2.4}$$

In which Count clip (n-gram) is the maximum number of n-grams co-occurring in a candidate translation and a reference translation, and Count(n-gram) is the number of n-grams in the candidate translation. To avoid very short translations which cause to maximize their precision rates, BLEU adds a brevity penalty (BP), to the Eq. 2.5:

$$BP = \begin{Bmatrix} 1 & if \;\; |c| > |r| \\ \in^{(1-|r|/|c|)} & if \;\; |c| \le |r| \end{Bmatrix} \tag{2.5}$$

$$BLEU = BP * \exp\langle \textstyle\sum_{n=1}^{n} W_n \, logP_n \rangle \tag{2.6}$$

The weighting factor, ***Wn***, is set at **1/*N***.

Scores are calculated over a whole test corpus — a set of sentences — by comparing scores with a set of reference translations while calculating these scores a grammatical adequacy or intelligibility are not principally taken into account. BLEU is designed to approximate human judgement at a corpus level, and performs poorly when has used to evaluate the quality of singular sentences [78]. In fact, a BLEU works with high order n-gram ***(n > 1)*** to aid candidate sentences with in sequence word matches and to assessment their fluency, a BLEU does not look at sentence level structure [79].

Table 2.4: The BLEU has been calculated for one source sentence with its possible three translations. The source sentence S = (s1, s2, ..., s|S|) has one ambiguous word, секіргенді. tst1 is output of RBMT, tst2 is output of Google translator and re*f* is a reference translation. The tst1 achieved better than tst2 of BLEU score 0.16%.

S	*Мен*	*акуланың*	*су*	*бетіне*	*секіргенді*	*көрдім*
ref	{Ben}	{köpekbalığının}	{su}	{yüzüne}	{ sıçrayarak}	{gördüm}
tst₁	{Ben}	{köpekbalığının }	{su}	{ yüzüne }	{ sıçradıgı}	{gördüm}
tst₂	{su}	{üzerinde}	{bir}	{köpekbalıgı}	{ sıçraması}	{gördüm}

The metric cannot equate via human judgements while ranking systems rely on different methods and is suggested for tracking improvements in performance over different configurations of the same system [80]. In addition, Denkowski et al. [81] state that the metric does not detect post-edition process that enhance translation quality. Instead, a BELU is over some benefits such as: it is speedy, language independent, and has been widely adopted.

In our experiments, instead of using human translation to make judgments, we evaluated machine translation's output by comparing it with its postedited text.

It is clear that ***tst₁*** is the best translation system because it shares many words with ***ref*** sentence. ***tst₂*** has achieved a modified unigram precision of **5⁄6** and ***tst₂*** achieves **2⁄6**. We have compared the result of the output from both of the MT RBMT and Google Translator.

3. KAZAKH–TURKISH MACHINE TRANSLATION

3.1 THE LANGUAGES

Kazakh is classified as a member of the Northwestern (or Kypchak) branch of the Turkic language family and is one of 4 members of the Kipchak-Nogai subcategory. Kazakh is most ethnically alike to Karakalpak, Crimean Tatar, and the Nogai languages. It is also inevitably intelligible with Kyrgyz, the native language of the neighboring Kyrgyz Republic. Kazakh is an official spoken language in Kazakhstan, where it is the national public language, sharing official status with Russian. They used Arabic script until 1929 and from 1929 into 1942-time interval is replaced with Latin alphabet. At the beginning of 1942 Kazakh like all other Turkic people are settled in the former Soviet Union had to use Cyrillic script which still in use today [82]. Large communities of native speakers are also surviving in China, neighboring Central Eurasian republics, Mongolia, Uzbekistan, and Russia. The entire number of speakers is at least 10 million people [82]. The present-day Kazakh Cyrillic alphabet consists of 42 letters, 33 of which are letters found in the Russian alphabet. There are controversial plans to transition to a Latin alphabet by 2025.

Turkish is classified as a member of the Southwestern (or Oghuz) branch of Turkic language family. With over 70 million L1 speakers [82], it is the Turkic language spoken by the most people. Modern Turkish is the predecessor of Ottoman Turkish. Ottoman Turkish had Arabic alphabet. Islamic influence also makes Ottoman Turkish absorb Arabic words into the language. Also, it had much more Persian words. After the founding of the Turkish republic in 1923, the language had changed by replacing Arabic script with Latin Script and it reformed its vocabulary of foreign elements. Today, the type of alphabet used in Turkish is Latin and it contains 29 letters. Nowadays it is spoken in different countries as Turkey, Northern Cyprus, and the Balkans, particularly in Bulgaria, Macedonia, Serbia, Romania, and Greece. Turkish considers the official language of Turkey and It is also, besides Greek, an official language in Cyprus. In addition, its closest relatives are Iraqi's Turkmen (spoken by less than 3 million people, mostly in Northern district), it has some relatives with Gagauz too (spoken by less than 200,000 people of Orthodox Christian religion, mostly in southern Moldova), Azerbaijanian (spoken by more than 20 million people in Iran and Azerbaijan) and Turkmen (spoken by some 3 million people in Turkmenistan) [83].

The differences between Kazakh and Turkish not consider a very big dissimilitude. This circumstance because Kazakh and Turkish trips (Kipchak, Oguz) for a long have been lived as neighbor. As it is known Oguz Turkic, before 1000 years had to live at the north of Seyhun (Sır Derya) Ridge (today's Kazakhstan territory) with Kipchak Turkic [84]. An MT system between Kazakh and Turkish has a big advantage for the language communities. MT can save time and money and also more secure than human translator.

3.2 MORPHOLOGY AND SYNTAX

3.2.1 Verbals

There are verbal tenses and moods common to both Kazakh and Turkish, like the definite past tense, the imperative mood, and the conditional mood. There are also quite a few differences. For example, Kazakh lacks the definite future tense affix **-{y}{A}c{A}k** known in Turkish; but has the so called *goal oriented future tense*, absent in Turkish: e.g., the Kazakh verb form *бармақпын* 'I intend to go' can be translated into Turkish as *gitmeyi düşünüyorum* 'I intend to go'. Another example of an affix found in one language but not in the other is the affix **-{D}{A}й** in Kazakh, which follows nouns and numbers and indicates resemblance, and can often be translated as the postposition gibi 'like' in Turkish.

Kazakh has several auxiliary verbs which are used for constructing analytic verbal forms. Four of them, the auxiliary verbs *жатыр, отыр, жүр, тұр* are used to construct the present continuous tense [85], as in the collocation *жауын жатыр* 'is raining', translated to Turkish as *yağıyor* 'is raining'. There are many other cases (i.e., not just due to analytic tenses) in which sequences of two or more Kazakh verbs map to a single verb in Turkish, as in the case of the *expression қуаныл кетті* 'gladdened', which is translated as *neşelendi* 'gladdened' in Turkish.

In the case of non-finite forms, there are one-to-many correspondences. For instance, Kazakh past verbal adjectives (participles) formed with the **-{G}{A}н** suffix can be translated into Turkish in at least three ways: as past verbal adjective with the **-m{I}**，**s** suffix, as a subject-relative verbal adjective formed with the **-{y}{A}n** suffix or as a past verbal adjective formed with the **-{D}{I}k** suffix. As an example, the Kazakh sentence *Сербия мен Қазақстан арасында шешілме**ген*** мәселе жоқ. ‘There aren’t any **unresolved** issues between Serbia and Kazakhstan’ can be translated into Turkish as *Sırbistan ve Kazakistan arasında, çözümlenme**miş** mesele yok.*, whereas the sentence *Екі мемлекет басшылары шағын және кеңейтіл**ген** құрамда келіссөздер жүргізді*. ‘The two leaders held talks in small and **expanded** format.’ in the parallel corpus we constructed is translated as *İki memleket başkanları kücük ve genişletil**diği** kapsamda müzekereler yönetti.*

Similarly, the Kazakh imperfect verbal adjective formed with the suffix **-{E}т{I}н** is translated as either a subject-relative verbal adjective formed with the **-{y}{A}n** suffix or as future verbal adjective constituted with **-{y}{A}c{A}k** suffix. For example, the Kazakh phrase *сөйлейтін* can be translated as *konuşacak* ‘which will speak’ or as *konuşan* ‘(which is) speaking’.

3.2.2 Nominals

Another example of a morphological difference between Kazakh and Turkish is the presence of a four-way distinction in Kazakh’s 2nd person system (both pronouns and agreement suffixes). In other words, in Kazakh there is a distinct word for all combinations of [±plural, ±formal] [85], whereas the Turkish 2nd person singular formal pronoun coincides with the 2nd person plural informal and 2nd person plural formal pronouns, as summarized in Table. 3.1 (both *siz* and *sizler* are used as the plural formal pronoun in Turkish).

Table 3.1: Second person personal pronouns in Kazakh and Turkish. Note the extra distinctions in the Kazakh forms.

	Kazakh		Turkish	
	-PLUR	+PLUR	-PLUR	+PLUR
-FRM	сен	сендер	sen	siz
+FRM	сіз	сіздер	siz	siz/sizler

3.3 SYSTEM

3.3.1 Morphological Transducers

The morphological transducers are based on the Helsinki Finite State Toolkit [86] – a free/open-source reimplementation of the Xerox finite-state tool chain, popular in the field of morphological analysis. It implements both the **lexc** formalism for defining lexicons, and the **twol** and **xfst** formalisms for modeling morphophonological rules. This toolkit has been chosen as it — or the equivalent XFST — has been widely used for other Turkic languages [87]- [91] and is available under a free/open-source license. The morphologies of both languages are implemented in lexc, and the morphophonologies of both languages are implemented in twol. The same lexc and twol files are used to compile both the morphological analyzer and the morphological generator for each language.

The Kazakh morphological transducer used in this work was presented in [84]. Turkish morphological transducer also comes from the Apertium project. It has not been described in a published work yet. Both transducers were extended to support all stems from the bilingual lexicon we constructed.

Table 3.2: Translation process (from Kazakh to Turkish) for the phrase *Айгүлдің мысығы үйде, ұйықтап жатыр*. ‘Aygül*’s* cat is sleeping at home’ some analyses are omitted for reasons of space. Note how a transfer rule has transformed a participle + auxiliary construction of Kazakh, *ұйықтап жатыр* ‘is sleeping’, to an analytic construction in Turkish, *uyuyor* ‘is sleeping’.

(Kazakh) Input	Айгүлдің мысығы үйде, ұйықтап жатыр
Mor. analysis	^Айгүлдің/Айгүл<np><ant><f><gen>$ ^мысығы/мысық<n><px3sp><nom>/мысық<n><px3sp><nom>+е<cop><aor><p3><pl>$ ^үйде/үй<n><loc>/үй<n><loc>+e<cop><aor><p3><pl>/үй<n><loc>+e<cop><aor><p3><sg>$ ^,<cem>$ ^ұйықтап/ұйықта<v><iv><prc_perf>/ұйықта<v><iv><gna_perf>$ ^жатыр/жат<vaux><pres><p3><pl>/жат<vaux><pres><p3><sg>$ ^.<sent>$
Mor. disambig	^Айгүл<np><ant><f><gen>$ ^мысық<n><px3sp><nom>$ ^үй<n><loc>$ ^,<cem>$ ^ұйықта<v><iv><prc_perf>$ ^жат<vaux><pres><p3><sg>$ ^.<sent>$
Lex. transfer	^Айгүл<np><ant><f><gen>/Ayg· ·ul<np><ant><f><gen>$ ^мысық<n><px3sp><nom>/kedi<n><px3sp><nom>$ ^үй<n><loc>/ev<n><loc>/mesken<n><loc>$ ^,<cem>$ ^ұйықта<v><iv><prc_perf>/uyu<v><iv><prc_perf>$ ^жат<vaux><pres><p3><sg>/<pres><p3><sg>/yat<v><iv><pres><p3><sg>$ ^.<sent>/.<sent>$
Structural transfer	^Aygul<np><ant><f><gen>$ ^kedi<n><px3sp><nom>$ ^ev<n><loc>$ ^,<cem>$ ^uyu<v><iv><prog>+i<cop><aor><p3><sg>$^.<sent>$
Mor. generation	Aygül’ün kedisi evde, uyuyor.

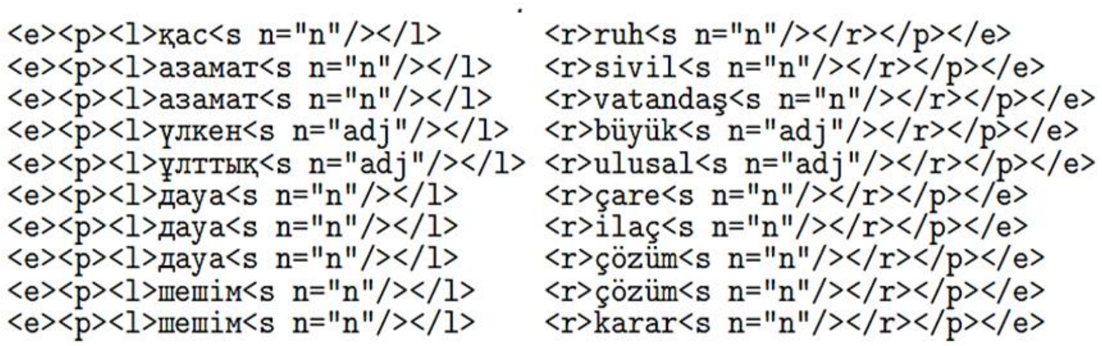
We decided to use the Turkish morphological transducer developed in the Apertium project and not the also free/open-source TRMorph [91], because the tagset used in the former is more consistent with morphological transducers developed in the Apertium project for other Turkic languages, including the Kazakh transducer. The consistency of the tagset allows to keep the transfer module relatively simple and pay more attention to the actual differences in the grammar of the languages rather than on differences in the tagset used.

3.3.2 Bilingual Lexicon

The bilingual lexicon currently contains 7,385 stem-to-stem correspondences and was built mostly by hand in the following way. We assembled a parallel Kazakh-Turkish corpus. For this we took all sentences from the Kazakh treebank [92] — approximately one thousand sentences — and translated them manually to Turkish. Then, these Kazakh and Turkish sentences were analysed with the **apertium-kaz** and **apertium-tur** morphological transducers. This provided the lemma and the part of speech tag for most of the surface forms in the corpora. The lemmas which were not already in the monolingual lexicons were added to them, and corresponding words were added to the bilingual lexicon. In addition, some of the stems present in the Kazakh lexc file but not found in the parallel corpus were translated into Turkish and added to the bilingual dictionary. Because of the similarity of the languages, the majority of entries in the bilingual dictionary (a file in an XML-based format) are one-to-one mappings of stems, but there are ambiguous translations. For example, the Kazakh word ‘азамат’ has two translations in Turkish: ‘sivil’ and ‘vatanda¸s’, as shown in Figure. 3.1.

3.3.3 Rules

The note made at the end of Section A.1 on replicability of the components aside, Apertium is primarily a rule-based MT system. Not counting morphophonology (morphotactics) rules required by HFST-based morphological transducers, there are three main categories of rules in our system — morphological disambiguation rules, lexical selection rules and transfer rules. A description of each follows



```
<e><p><l>қас<s n="n"/></l>       <r>ruh<s n="n"/></r></p></e>
<e><p><l>азамат<s n="n"/></l>    <r>sivil<s n="n"/></r></p></e>
<e><p><l>азамат<s n="n"/></l>    <r>vatandaş<s n="n"/></r></p></e>
<e><p><l>үлкен<s n="adj"/></l>  <r>büyük<s n="adj"/></r></p></e>
<e><p><l>ұлттық<s n="adj"/></l> <r>ulusal<s n="adj"/></r></p></e>
<e><p><l>дауа<s n="n"/></l>      <r>çare<s n="n"/></r></p></e>
<e><p><l>дауа<s n="n"/></l>      <r>ilaç<s n="n"/></r></p></e>
<e><p><l>дауа<s n="n"/></l>      <r>çözüm<s n="n"/></r></p></e>
<e><p><l>шешім<s n="n"/></l>     <r>çözüm<s n="n"/></r></p></e>
<e><p><l>шешім<s n="n"/></l>     <r>karar<s n="n"/></r></p></e>
```

Figure 3.1: Example entries from the bilingual lexicon. Kazakh is on the left, and Turkish on the right. Each stem is accompanied by a part-of-speech tag and there may be many–many correspondences between the stems.

3.3.4 Morphological Disambiguation Rules

The system has a morphological disambiguation module in the form of a Constraint Grammar (CG) [93]. The version of the formalism used is vislcg. The goal of the CG rules is to select the correct morphological analysis when there are multiple analyses. We used the Kazakh CG previously developed partially by the authors of this paper and partially by other Apertium contributors. At the time of this writing the file contains 164 rules. Due to closeness of the languages, the majority of ambiguity may be passed through from one language to the other.

3.3.5 Lexical Selection Rules

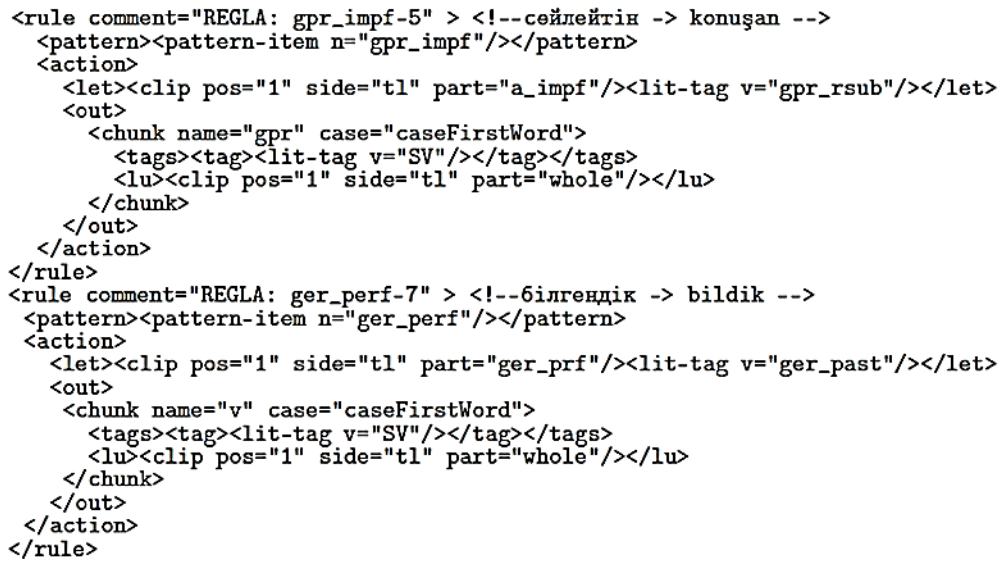
In general, lexical selection rules are necessary to handle one-to-many correspondences of the bilingual lexicon. While many lexical items have a similar range of meaning, lexical selection is sometimes necessary when translating between Kazakh and Turkish as well. For example, the Kazakh word *am* has two meanings: *am* ‘name’ and *am* ‘horse’ and can be translated into Turkish as either *ad* ‘name’ or *at* ‘horse’. A lexical selection rule chooses the translation at ‘horse’ if the immediate context includes a word *ұста* ‘hold’. Another example is the word *жарық*, which as a noun can mean either ‘light’ or ‘crack’. It is translated to Turkish by default as *ışık* ‘light’, and is translated as *yarık* ‘crack’ only in the immediate context of words like *есік* ‘door’ and *қабырға* ‘wall’. A relatively small number of 92 lexical selection rules were developed and added to the system. The lexical selection module we used [97] allows inferring such rules automatically from a parallel corpus, but we have not employed this feature of it yet.

3.3.6 Structural Transfer Rules

Apertium, as a rule, translates lemmas and morphemes one by one. Obviously, this does not always work, even for closely related languages. Structural transfer rules are responsible for modifying morphology or word order in order to produce “adequate” target language.

As seen in Table 3.2, the structural transfer module takes a sequence of (source language lexical form — target language lexical form) pairs in the following format: **ˆSL-lemma<SL-tag1><SL-tag2> <...><SL-tagN> /TL-lemma<TL-tag1><TL-tag2><...><TL-tagN>$** TL lemma and tags are provided by the preceding two modules — lexical transfer and lexical selection. The lexical transfer module looks up the TL lemma and usually the first one or two tags (read: part of speech tag) in the bilingual transducer, the rest of the tags are carried over from the SL.

Figure 3.2 gives an example of a transfer rule. Any transfer rule consists of two core parts — of a pattern and an action. The pattern exemplifies fixed length sequence of source language lexical form (SLLF). It is sole of concatenation of lexical form which may require a combined processing additional to the straightforward word-for-word translation, due to the grammatical disparity between SL and TL (gender and number changes, reorderings, prepositional changes, etc) [95]. On the other hand, the action consists a bunch of “commands” that have to be implemented to process as requested every matched pattern. It is achieved on SLLF are determined, and the TL pattern is built. In this case, the pattern named “gpr_impf” matches Kazakh verbal adjectives formed with the**-{E}т{I}н** affix. Recall from Section 3.2.1 that Kazakh verbal adjectives ending in **-{E}т{I}н** have two possible translations in Turkish — either with a verbal adjective ending in **-{y}{A}n** suffix or with a verbal adjective ending in **-{A}c{A}k** suffix. The rule in Figure 3.2 replaces the **<gpr_impf>** tag on the TL side with the **<gpr_rsub>** tag, which corresponds to a **-{y}{A}n** verbal adjective in Turkish. It is important to mention that, if a sequence of lexical forms matches two different rules, firstly, the longest is chosen, and secondly, for rules of the same length, the one defined before is chosen. In addition, transfer rules perform chunking, and later transfer stages can operate on chunks of words as if they were single words, but we will not discuss chunking-based rules here since this technique is currently not employed in the Kazakh-Turkish translator.



```
<rule comment="REGLA: gpr_impf-5" > <!--сейлейтін -> konuşan -->
  <pattern><pattern-item n="gpr_impf"/></pattern>
  <action>
    <let><clip pos="1" side="tl" part="a_impf"/><lit-tag v="gpr_rsub"/></let>
    <out>
      <chunk name="gpr" case="caseFirstWord">
        <tags><tag><lit-tag v="SV"/></tag></tags>
        <lu><clip pos="1" side="tl" part="whole"/></lu>
      </chunk>
    </out>
  </action>
</rule>
<rule comment="REGLA: ger_perf-7" > <!--білгендік -> bildik -->
 <pattern><pattern-item n="ger_perf"/></pattern>
 <action>
   <let><clip pos="1" side="tl" part="ger_prf"/><lit-tag v="ger_past"/></let>
   <out>
    <chunk name="v" case="caseFirstWord">
      <tags><tag><lit-tag v="SV"/></tag></tags>
      <lu><clip pos="1" side="tl" part="whole"/></lu>
    </chunk>
   </out>
 </action>
</rule>
```

Figure 3.2: Examples of a transfer rule. The first rule translates Kazakh -{E}т{I}н verbal adjectives with -(y)An verbal adjective by replacing the <gpr_impf> tag on the TL side with the <gpr_rsub> tag. The second rule transfer Kazakh -{G}{A}нл{I}{K} verbal noun (gerund) into -{D}{I}k verbal noun (Gerund) by replacing the <ger_perf> tag on the TL side with the <ger_past> tag.

3.4 EXPERIMENTS

3.4.1 Translation Quality

The translation quality was measured using two metrics, the first was word error rate (WER), and the second was position-independent word error rate (PER). Both metrics are based on the

Levenshtein distance (Levenshtein, 1965). Metrics based on word error rate were chosen as to be able to compare the system against systems based on similar technology, and to assess the usefulness of the system in a real setting, that is of translating for dissemination. Besides calculating WER and PER for our Kazakh-Turkish MT system, we did the same for two other publically available Kazakh-Turkish MT systems — from Google Translate and Yandex Translate. The procedure was the same for all three. We took a small (1,025 tokens) Kazakh text, which was a concatenation of several articles from Wikipedia and translated it using the three MT systems. The output of each system was postedited independently to avoid biasing in favour of one particular system. Then we calculated WER and PER for each using the **apertium-eval-translator** tool and BLEU using the **mteval-v13a.pl** script. Note that BLEU score is typically calculated by comparing against a pre-translated reference translation, where here we calculate against posteditted reference translations for each of the systems.

3.4.2 Results

Table 3.3 shows the results obtained for all three systems — Google, Yandex and Apertium. Google's lowest WER. Apertium has a comparable WER despite having much higher number of Previous and related work words. Yandex Translate's WER is higher, but PER is similar to the other two. These numbers can be compared with scores for other translators based on the Apertium platform. For example, the Kazakh–Tatar system described in [20] achieves post-edition WER of 15.19% and 36.57% over two texts of 2,457 words and 2,862 words respectively. The Tatar–Bashkir system in [21] reports WER of 8.97% over a small text of 311 words and WER of 7.72% over another text of 312 words. The higher word error rate can be explained by the fact that Kazakh and Turkish are more distantly related than Tatar and Kazakh or Bashkir.

Table 3.3: Word error rate and Position-independent word error rate; OOV is the number of out-of-vocabulary (unknown) words. The Google system has a similar word error rate to the Apertium system despite the significantly lower number of out-of-vocabulary words. Note that the BLEU scores are computed against a postedited reference translation.

System	OOV	WER (%)	PER (%)	BLEU
Yandex	43	69.73	48.63	2.84
Apertium	128	45.77	41.69	16.67
Google	5	43.85	33.67	16.32

4. TRANSFER RULE APPLICATION AND RULE LEARNING

4.1 MAXIMUM-ENTROPY STRUCTURAL TRANSFER

Linguistic classification problems consider as main problems in natural language processing, in which linguistic class is predicted by linguistic context. Maximum entropy is powerful machine learning algorithm that is useful for solving these problems by combining different portions of contextual evidence so that to consider the probability of a definite linguistic class appearing with a definite linguistic context. Likewise, the maximum entropy approach has used by SMT to predict target-side for an ambiguous source-side. It applies the maximum entropy principle to a probabilistic model of transfer rules. On the other words it is estimating a natural language model based on the maximum entropy [96]. One of the important aspect of the maximum entropy is integrating a rich contextual information as features and this can aid SMT systems to perform context-dependent rule selection. The features defined as binary feature function (expresses statistical properties of a language model) $f \mid X \_ Y \rightarrow \{0, 1\}$ which divide $f\mid X \_ Y$ into two subsets. In this case the probability of a translation t being the translation of a word s in an SL context c be $ps(t|c)$. However, for each combination of (s, t, c), the probability of a translation could be computed directly from the available corpora, but there are couple of restriction. First of all, how could relevant contexts be chosen. And how the maximum entropy deals with the translations of words which are not even exist in the corpus. The maximum entropy model will overcome all the mentioned restrictions. For the first restriction the maximum entropy just allows the contexts which are linguistically relevant to be defined a priory and then integrate these smoothly into a probabilistic model [97] and the restriction will rise because the unknown words, the maximum entropy will not estimate about what is not exist in the training data. Therefore, when these is not any information in the training data, the maximum entropy expect that all outcomes are equally likely. The assumption of the maximum entropy has been applied into the problem of lexical selection within apertium platform [98] to project the problem of lexical selection in statistical MT like a classification problem. That was realized with learning an independent maximum-entropy classifier for every SL word, using SL context to portion between possible translations.

In the classifier each feature is assigned a weight λ_s through the training process. And the probability of a translation ***t*** for word ***s*** in context can be seen after integrating the assigning weights as in the Equation. 4.1. Notice that ***Z*** is a normalizing constant. Thus, the most probable translation can be found using the equation 4.2.

$$p_S(t|c) = \frac{1}{Z}\, exp \sum_{k=1}^{nf} \lambda_{k=1}^{s} n_k^s(t|c) \tag{4.1}$$

$$\underset{t\epsilon T_{(s)}}{argmax}\, p_S(t|c) = \underset{t\epsilon T_{(s)}}{argmax} \sum_{k=1}^{nf} \lambda_{k=1}^{s} n_k^s(t|c) \tag{4.2}$$

The ***argmax*** operation designates the search problem, i.e. the production of the output sentence in the TL.

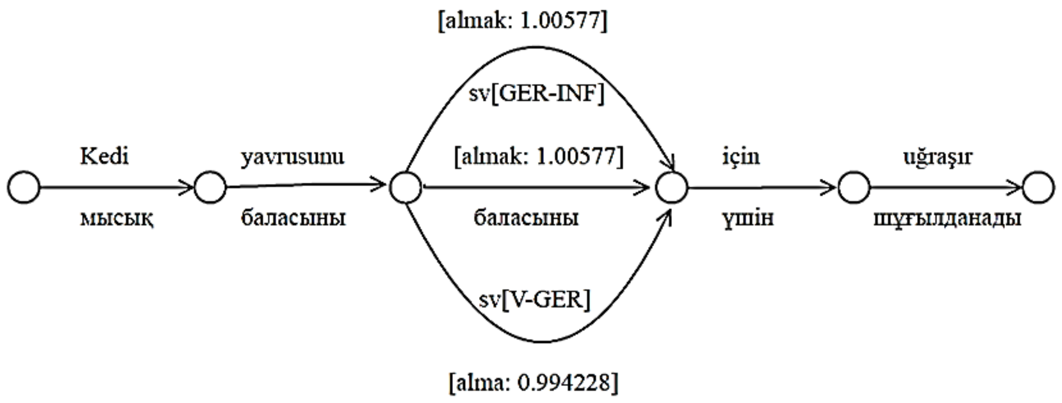


Figure 4.1: The maximum-entropy approach for a rule-application process. The weights are gathered for every translated word of each possible target sentence and when a position in each target sentence comes when where ***q0*** is the only alive state, the translation with the highest combined weights all the chosen. For the words *алып кету* ‘take’ the translations almak and alma is chosen, the translation of sentence *мысық баласыны алып кету үшін шұғылданады* ’Cat gives labor to get her baby‘ is kedi yavrusunu almak için uğraşır 1.0057 is the highest weight over other translations

The features that we interpret are similar to a subset of the structural-rules-selection which was described in Chapter 2. The whole structure of the direct maximum entropy models is summarized in the Figure. 4.3.

4.1.1 Rule Application

The process of applying rules materialize after getting the lexcical analysis of the input sentences by applying the rest of the apertium tools biltrans and lexctor respectively. The lexical analysis of the input sentences is a string of tokens (words) each with its translations and part of speech tags. We will split these strings into source and target tokens along with their tags. After that the process of applying rules is conducted in the following steps: First, we will match these tags with categories from the transfer file as these matches will help us match the tokens to the transfer rules. Second, transfer rules function is matching categories with pattern items of transfer rules and applying a sequence of actions. There could however, be more than one potential sequence of actions for each SL pattern.

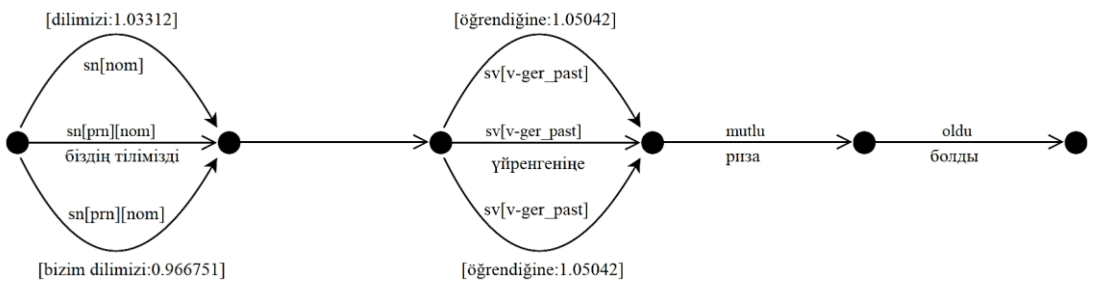


Figure 4.2: The maximum-entropy rule-application process. The weights for each translation are summed, and the translation with the highest combined weights is picked. For the words *біздің тілімізді* ‘our language’ *үйренгеніңе* ‘to learn’ the translations *dilimizi öğrendiğine* is chosen, the translation of sentence *біздің тілімізді үйренгеніңе риза болды* ‘she/he was happy to learn our language’ is *dilimizi öğrendiğine mutlu oldu*. 2.08354+1.05042=3.13396 is chosen over *bizim dilimizi öğrendiğine mutlu oldu*. 0.966751+1.05042=2.017171

Next, the probability $\boldsymbol{P_s (t|c)}$ is computed for all active features (rules), Table 4.3 presents the SL sentences, and output of the possible structural transfer rule-selection paths after scoring on a target-language model. The scores are normalized as a fractional count. We sum up the weights of the live rules for each TL translation of each SL word in place of selecting the longest rule using Equation. 4.1. Notice that the process of summing the weights of the live rules the for each TL translation of each SL word and choosing the heaviest rules carry out with Beam search algorithm. More explanation of how we execute the Beam Search described in the section 4.2. When the initial state q0 is the only living state in the transducer, we retract and pick the translation with the highest total of weights as declared in Figure. 4.2. and Figure 4.1 in turn. However, when there are more than one rules matching the same pattern, this will cause ambiguity. And if many patterns have ambiguities this make the whole sentence has much ambiguities, as all the possible combinations will be generated by applying different rules on same pattern of the SL sentences, these the possible combinations are equal the multiplication of each number of ambiguous rules of each pattern.

In this case, the output will be all the possible combinations of translations of the sentence along with their analysis (output of the rules), but for some sentences we had millions of combinations (15 words for each 3 ambiguous rules would make about 14 million combinations) , and that cause memory problems and accuracy problems in scoring that so many combinations by the language model.

We track another approach rather than generating all possible combinations, we choose the left-right-longest match rule for all ambiguous patterns except the first ambiguous pattern. In other words, we just generate all possible combinations for the first ambiguous pattern and choose left-right-longest match rule for other patterns in one sentence. Thus, in place of choosing the longest rules for first word, we sum up the weights of active rules for a SL word (assign to Equation 4.1). Eventually, we perceive the translation with height sum of weights. Previously we come to a position in the sentence where the only viable state in the transducer is the beginning state ***q0***, we return back and choose the highest sum of weights of that translation.

The comprehensive structure of the direct maximum entropy models is outlined in Figure. 4.3. It should be noted that the main reason for using maximum entropy despite scoring our sentences with TLM, is that maximum entropy indicates which features are relevant and how to weight them. This means that we do not have to carry out all translations, but if we only use the language model, we should accomplish all the translations before scoring them. This saves time and increases performance.

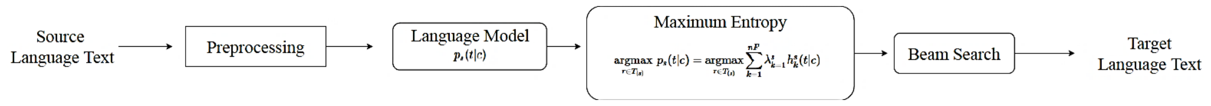

Source Language Text
Preprocessing
Language Model
$p_s(t|c)$
Maximum Entropy
$\underset{r \in T_{|s)}}{\operatorname{argmax}}\, p_s(t|c) = \underset{r \in T_{(s)}}{\operatorname{argmax}} \sum_{k=1}^{nF} \lambda^s_{k=1} h^s_k(t|c)$
Beam Search
Target Language Text

Figure 4.3: Architecture of the translation approach based on maximum entropy models. In addition, the maximum entropy approach has various benefits such as we can easily extend by adding more features, and it is possible to incorporate into different SMT models

In this work we have tried to extend the maximum entropy by adding more features, the section 4.4 gives more details about this part. Besides, in the Figure 4.4 and the Figure. 4.5, we explain the whole structure of training and testing stages.

4.2 TRANSLATION USING BEAM-SEARCH

Beam search is a fast and empirically effective method for translation, it is central to largescale MT systems because their efficiency and tendency to produce high-quality translations ([102, 74, 103]. We use a fundamental version of the beam search algorithm to find a translation that maximizes the conditional probability accorded by the maximum entropy model. First, we apply a set of ambiguous rules to some words, and then we obtain the weights of these words for every rule from models of the maximum entropy. Thereafter, we build a tree for these new words. The tree is based on vectors of rule indices along with the sum of their weights. Let us assume, that at any iteration, we have a set of rules applied to the same words. The beam tree would expand with the number of rules applied to the words. For example, if in any iteration a set of rules exists ($\boldsymbol{R}$ **= 5** rules and $\boldsymbol{W}$ **= 3** words) in which every word matches three rules, three different translations are obtained for each word. In this case, the beam tree is expanded to have nine translations, and these trees merge with the existing beam tree. Then, we sort those nine translations by descending sum of weights, and reduce the beam tree to have no more than the beam size translations. Supposing that the beam size equals four, we remove the last five translations from the tree, and continue until we finish all the ambiguous rules, and the output is a tree with no more than the beam size translations. Finally, we obtain and output only the best translation.

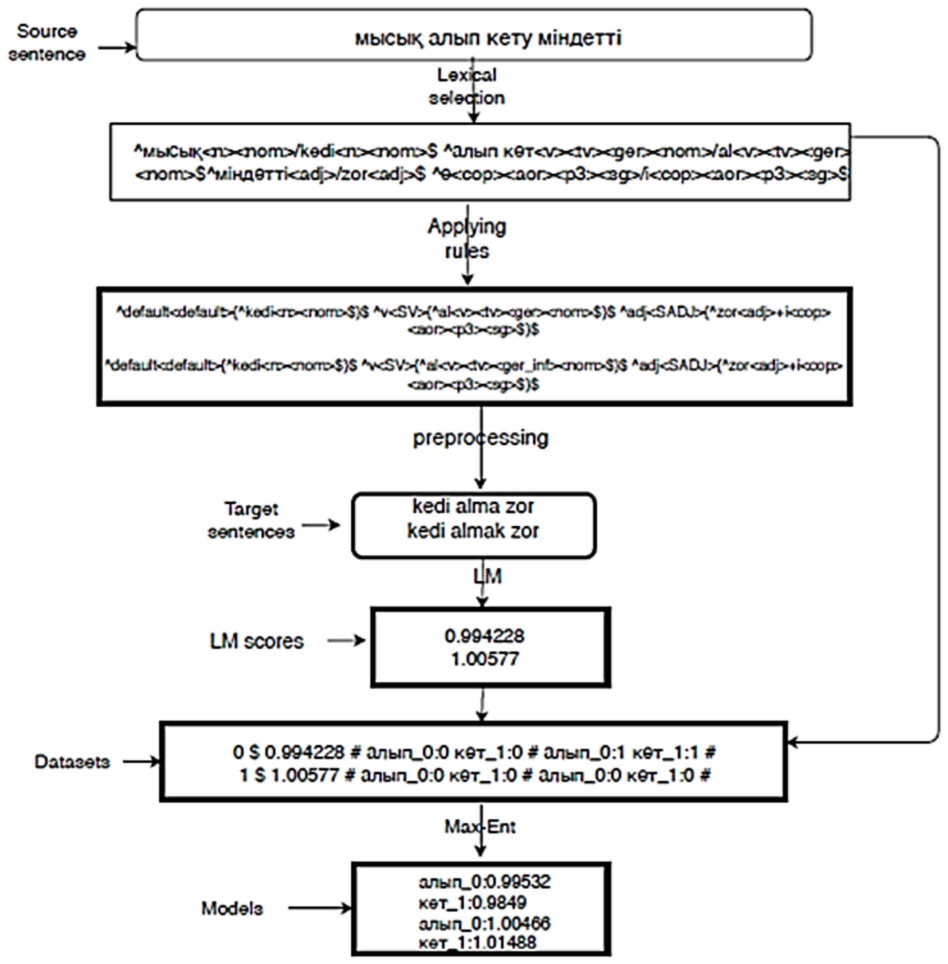

Source sentence
мысық алып кету міндетті
Lexical selection
^мысық<n><nom>/kedi<n><nom>$ ^алып кет<v><tv><ger><nom>/al<v><tv><ger><nom>$^міндетті<adj>/zor<adj>$ ^е<cop><aor><p3><sg>/i<cop><aor><p3><sg>$
Applying rules
^default<default>{^kedi<n><nom>$}$ ^v<SV>{^al<v><tv><ger><nom>$}$ ^adj<SADJ>{^zor<adj>+i<cop><aor><p3><sg>$}$
^default<default>{^kedi<n><nom>$}$ ^v<SV>{^al<v><tv><ger_inf><nom>$}$ ^adj<SADJ>{^zor<adj>+i<cop><aor><p3><sg>$}$
preprocessing
Target sentences
kedi alma zor
kedi almak zor
LM
LM scores
0.994228
1.00577
Datasets
0 $ 0.994228 # алып_0:0 кет_1:0 # алып_0:1 кет_1:1 #
1 $ 1.00577 # алып_0:0 кет_1:0 # алып_0:0 кет_1:0 #
Max-Ent
Models
алып_0:0.99532
кет_1:0.9849
алып_0:1.00466
кет_1:1.01488

Figure 4.4: Training process

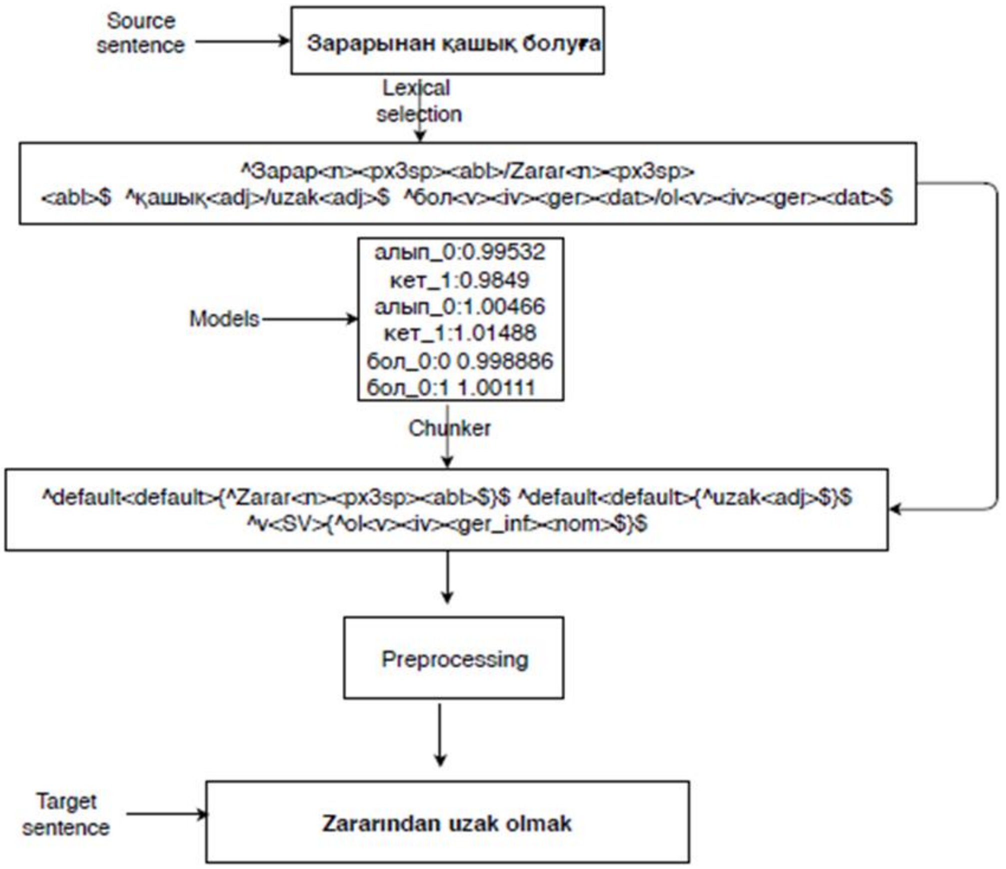

Source sentence
Зарарынан қашық болуға
Lexical selection
^Зарар<n><px3sp><abl>/Zarar<n><px3sp><abl>$ ^қашық<adj>/uzak<adj>$ ^бол<v><iv><ger><dat>/ol<v><iv><ger><dat>$
алып_0:0.99532
кет_1:0.9849
алып_0:1.00466
кет_1:1.01488
бол_0:0 0.998886
бол_0:1 1.00111
Models
Chunker
^default<default>{^Zarar<n><px3sp><abl>$}$ ^default<default>{^uzak<adj>$}$ ^v<SV>{^ol<v><iv><ger_inf><nom>$}$
Preprocessing
Target sentence
Zararından uzak olmak

Figure 4.5: Testing process

Table 4.1: Statistics of the test corpora and the training parallel corpora used by three systems (Weighted, NMT, and Moses). The columns SL and TL present the total number of tokens in the source and target languages, and the column ″Dev″ signifies the number of sentence pairs used as development data by NMT and SMT. The columns ″No.amb″ and ″% am-big″ indicate the number of words with multiple translations and the percentage of SL words that have multiple translations, respectively.

	Lines	SL	TL	DEV	No. amb	%am-big.
Train	62,893	266,555	285,648	5000	9.999	-
Test	1000	9,158	9,249	-	1,619	5.65

4.3 EXPERIMENTS

We have learned rules and weights as binary features, for this we have used the implementation of generalized repetitive scaling available in the YASMET for counting the feature weights. We evaluate the system with the same evaluations metrics mentioned in the previous chapter. Furthermore, the system was evaluated and compared with state-of-art MT systems, in which we used the same test data from testing the weighted system and comparing it with reference systems. The whole information about the test and training data presented in the Table 4.1 and we compared the output of all systems with postedited version of our weighted system. We did not postedit each system independently because when we checked the output of other baseline systems, such as the Moses system, we found that most of the words are not in the vocabulary as presented in Table 4.5 and more than %50 of words is unknown and some sentences are not fully generated as target sentences.

Regarding the NMT system, the output (target) sentences are not related to the input sentences (source) sentences, which means that many of the sentences are translated arbitrarily and out of context. Nevertheless, the output sentences are not fully generated. Owing to the errors above, postedite the output of each system independently is not be a good decision, and would be more detrimental than being beneficial to the system performance. Therefore, we decided to postedit our weighted system and evaluate all of the other baseline systems to calculate the error rate of produced text by using translation quality measurement method for all the systems compared with the postedited version of our weighted system.

Reference systems

- Linguistic defaults. This is a translation which is described as default translation, it is a left-right longest match algorithm's outcome, which means there is more than one rule apply into the same pattern. Then the translation of applying left-right-longest match rule will be chosen, this translation also considers the most general translation.

- Linguist-chosen random. When structural transfer rules have a similar pattern's name, that mean we can obtain more than one translation for one sentence, and here we are just going to choose randomly one of these translations.

- Target language model. After using one of structural transfer method for getting all possible translations from our system, we used language model method on-line to score these target sentences. The translation with supreme score has chosen as best translation. This is the method used by [73]. We used six-grams language model to score the generated TL side. This system has given a very efficient result compared with other systems.

The n-grams around an ambiguous word will be consider through the learning context. An n-grams is an adjoining sequence of words. The Table 4.2 is a good illustration of n-grams extracted form a sentence. The example in the Table 4.2, we just take 1-5 grams around ambiguous pattern because the sentence only contains five patterns.

The purpose of this method is to produce SL n-grams which involve an ambiguous word in which every ambiguous word is annotated with its translation. After that, we calculate how many times each translation has to be appears along with each n-gram, we then generate a rule that choose the most common translation of the source-language word in a particular n-gram context. Here, Tyers et al. [94] is noted that, for working with this method, it should be useful to know the rules will benefit the current module, because not all rules generated are adequate.

4.3.1 Results and Discussion

The weights of binary features are calculated by using the execution of generalized repetitive scaling available in the YASMET tool to calculate feature weights.

Evaluation results are presented in a Table 5.8, which compares the outcomes for the new approach (weighted) with the default (unweighted), randomly selected, and results are acquired by using the TL model online, for the language pair in Apertium with our two common evaluation metrics. In addition, no large difference exists in the evaluation results because we selected the test data randomly from corpora of different articles, and not all sentences have ambiguous words.

Furthermore, in some ambiguous words, the unweighted achieved a performance equal to that of the weighted. Significant enhancement with respect to TL model performance is expected as a result of the impressive application that the maximum-entropy model executes of information regarding appropriate SL contexts and their translations, during the weighting of features that represent those SL contexts over the entire corpus.

Table 4.2: The 1 to 5grams around the word *алып кету alma, almak* ’take’ for a sentence in Kazakh. The brackets indicate the borders between particular n-grams for each value of n.

“мысық баласыны алып кету үшін шұғылданады”	
n	n-gram
1	[алып кету]
2	[баласыны алып кету][алып кету үшін]
3	[мысық баласыны алып кету][баласыны алып кету үшін][алып кету үшін шұғылданады]
4	[мысық баласыны алып кету үшін][баласыны алып кету үшін шұғылданады]
5	[мысық баласыны алып кету үшін шұғылданады]

In addition, to evaluate our system on the parallel corpus and compare it with the performance of state-of-the-art MT systems trained on the same corpora, we trained the NMT and SMT baseline systems as the weighted system by taking the parallel dataset from the KDE4 corpus . Table 4.5 exhibits the performance of the weighted system.

First, we compared the results of our weighted system with NMT , an NMT-small model from OpenNMT, with a framework employing neural translation. We trained the model at word level by using byte-pair encoding second, we compared the weighted system with other publicly available SMTs such as the Moses system . We used a phrase-based decoder in the Moses system [71], which allows us to create phrase-based systems using standard features that are usually used in current systems. The phrase-based decoder is used to train translation models for our language pair. Additionally, we trained 3-gram language models with Kneser–Ney smoothing using KenLM [101].

As shown in Table 4.5, the performance of the weighted system is much better than that of other baseline systems; the weighted system established a baseline of WER 41.78, PER 40.13, and BLEU 31.20. For the out-of-vocabulary (unknown words) coverage in the corpus that we used for our experiments, the weighted system outperformed the NMT and Moses systems in WER, PER, and BLEU. One reason for the results is that the orthographic and dialectal variety of the texts used in the aligned corpus, may have prevented learning and generalization in the SMT and NMT systems. The weighted (RBMT) system is able to overcome this issue to some degree. Adding variants of frequent words is a simple issue, and one that we frequently addressed while developing the weighted system on the Wikipedia and news corpora.

The evaluation results of NMT were insufficient compared with our weighted system. Table 4.5 shows a very low BLEU score of 0.05, very high WER score of 96.85, and PER score of 92.26, which were obtained through our experiments in the NMT. However, most errors for the NMT system can be a factor in this event. Some sentences were much longer than the average appropriate length for NMT, thereby resulting in poor translation because encoder-decoder NMT models were unable to translate long sentences.

Table 4.3: The Kazakh–Turkish monolingual corpus. The table introduces the SL sentences and output of all the possible structural transfer rule-selection paths after scoring on a target-language model. The fractional counts have been obtained by normalizing the scores.

S		Sentence	$p(\boldsymbol{g_i}\|s)$
S_1		Айжанның үй жұмысын дайындағанына анасы өте қуанды	
	$T(g_1, S)$	Ayjan’ın ödevi hazırladığına anası çok neşelendi	0.025393
	$T(g_2, S)$	Ayjan’ın ödevi hazırlayarak anası çok neşelendi	0.026263
S_2		Ол интернетте көрген суреттерге күлгеніңді жақсы білды	
	$T(g_1, S)$	O internette gördüğü resimlere gördüğünü epey bildi	0.026263
	$T(g_2, S)$	O internette gören resimlere gördüğünü epey bildi.	0.026937
	$T(3, S)$	O internette gördüğünü resimlere gördüğünü epey bildi	0.026179
	$T(g_4, S)$	O internette gördüğünü resimlere gülerek epey bildi	0.026937
S_3		Оқушы сабақ ұққанына бес минут болды	
	$T(g_1, S)$	Okuyucu ders duyduğuna beş dakika oldu	0.027609
	$T(g_2, S)$	Okuyucu ders duyarak beş dakika oldu	0.035838
S_4		Бала қыздың ұялғанына сенбеді	
	$T(g_1, S)$	Okuyucu ders duyduğuna beş dakika oldu	0.036681
	$T(g_1, S)$	Okuyucu ders duyarak beş dakika oldu	0.040048
S_5		Алпамыс өз уақытын жоспарлағанды дұрыс деп санайды	
	$T(g_1, S)$	Alpamys kendi vaktini planladığı doğru diyip sanar	0.052628
	$T(g_2, S)$	Alpamys kendi vaktini planlayarak doğru diyip sanar	0.023334
S_6		Мұғалім оқытқанды ұмытып кетті	
	$T(g_1, S)$	Derste hocadan soru sorma zor.	0.022880
	$T(g_2, S)$	Derste hocadan soru sormak zor	0.035261
	$T(g_3, S)$	Derste hocadan soru sorma zor	0.035243
	$T(g_4, S)$	Derste hocadan soru sorma zor	0.035261

The NMT system performed poorly on lengthy sentences, but is relatively good up to a sentence length of approximately 60 words. As the NMT system produces short translations (length ratio 0.859, opposed to 1.024), the quality of these translations is drastically low [102]. Moreover, lack of data is a main reason for the poor quality of the NMT system. The figures obtained, given approximately 265,000 tokens of training data on each side seem to be consistent with experiments conducted on the relation of NMT performance and the amount of data [102]. Another reason for the poor performance was the relative lack of language standardization. Furthermore, the NMT system exhibited worse translation quality out of domain than normal, which is a familiar challenge in translation in a different domain. In NMT, a domain can be described by a corpus from a specific source, and may diverge from other domains in topic, genre, style, level of formality, and other factors. Input words have various translations and their meanings are predicated in different styles. NMT is adapted for the sake of fluency. Although the output of the NMT system is sufficiently fluent, it is still completely unrelated to the input. Most of the errors in the weighted system are due either to mistakes and gaps in the morphophonology components and disambiguation errors or input words being out of vocabulary.

Furthermore, lexical selection was one of the causes of errors. The reason for this was that we made our system to select the first translation of an input word when more than one translation of the input word existed, and the first translation was not always suitable. The test corpus used for evaluation was not used while developing the RBMT system, including the training and development sets.

In case of SMT, we achieved a BLEU score of 1.33, WER score of 91.04, and PER score of 85.87 respectively. These results are lower than those of our weighted system. The error causes include the following reasons:

Table 4.4: Word error rate and Position-independent word error rate; OOV is the number of out-of-vocabulary (unknown) words. WER and PER scores with 75% confidence intervals for the reference

systems on the test corpora. Note that the BLEU scores are computed against a *postedited* reference translation.

System	OOV (%)	WER (%)	PER (%)	BLEU
Weighted	**0.28**	**27.72**	**26.51**	**53.28**
Unweighted	0.28	32.14	31.88	42.28
TLM	0.28	28.85	28.48	48.31
Random	0.28	30.91	30.00	44.73

Table 4.5: Word error rate and Position-independent word error rate; OOV is the number of out-of-vocabulary (unknown) words. WER and PER scores the reference system on the test corpora. Note that the BLEU scores are computed against a *posteditted* reference translation.

System	OOV (%)	WER (%)	PER (%)	BLEU
Weighted	17,2	43.91	42.13	31.20
NMT	1,92	98.77	96.56	0.05
Moses	58,00	98.77	90.13	1.33

The main error category is a factor of the scarcity of data used during our experiments. Furthermore, the performance declines with a limited amount of parallel data. Big data is expected to yield better performance. Another reason was the existence of low-frequency words and word formation errors, which characterize the morphological richness of Kazakh and Turkish, and that negatively affect the quality of the SMT system; this system performs poorly on morphologically rich languages [102]. In addition, we found that the translation was less fluent. Some errors were related to accuracy, particularly mistranslation and omission.

4.4 FURTHER EXPERIMENTS FOR RULE–LEARNING APPROACH BY MAXIMUM ENTROPY AND COMPARING REFERENCE SYSTEMS

The maximum entropy as it is aforementioned above, has ability to integrate a rich contextual information as features, this can make SMT systems accomplish context-dependent rule selection. The features defined as binary feature function (expresses statistical properties of a language model) $\boldsymbol{f \mid X \_ Y \rightarrow \{0, 1\}}$ which divide) $\boldsymbol{f \mid X \_ Y}$ into two subsets. And based on this view, we try to add more contextual in*f*ormation as features to evaluate the performance of maximum entropy. These features represent tags and lemmas. Tags are unambiguous grammatical categories to words in a text and lemmas is a base form of words, during our experiments we added more features as much as possible. In Figure 4.10, we depict the evaluation performance of three version of the weighted system. First, in case of the weighted system's Models with unknown words and extra feature (MBT), these models with words of bad sentences and adding more contextual information as features or as tags. The bad sentences mean the sentences with unknown words or with words have been attached with the special characters such as ('*', '@' and '#') and these characters means the unanalyzed words, untranslated lemma, and in morphological generation, the module is not able to generate the surface form of the words respectively. We make the MBT weighted system learn the weights of rules based on these properties. Second, in the models without unknown words and with extra feature (MNBT) of the weighted system, the models have been obtained after adding more contextual information as features and removing all whole sentences with the unknown words or the words attaching with special characters ('*', '@' and '#') form training corpus.

We execute the MNBT weighted system to prevent breaking transfer rules during applying these rules on input words when there are unknown words. Third, the last version of the weighted system with models without unknown words and extra feature (MNBNT), the models of this system without unknown words and additional features. Here, we have trained the MNBNT to learn the weights of rules without any broken through applying rules to input sentences. It would be noted that, the process of eliminating sentences with unknown words or words sticking with special characters have caused a huge decline in the size of training corpus. Though, eliminating sentences with unknown words will prevent the breaking of transfer rules while applied to the input sentences, depending on our experiments during this work a small size of training data can affect badly the performance of learning algorithm which means system cannot learn better with tiny in size data. However, we achieve the process of eliminating bad sentences from training corpus to just see how system performance recruit in general when there are not any bad sentences in a whole corpus.

Consequently, we have compared the test result of three versions of the weighted system (MBT, MNBT, and MNBNT) with each other and with the original weighted system (system models with unknown words and only processing the first tag from multiple tags in the sentences analysis). Notice that, with the all three systems, we have used the same test data which is used previously in the preceding chapters. Moreover, we evaluated the results of three systems by using the same evaluation metrics as aforementioned before which are WER metric, PER metric, and BLEU metric. The three systems (MBT, MNBT, and MNBNT) could not outperform the original weighted system, even though we added additional features and deleted the bad sentences of three systems. The results are stable for the system with models (MBT, MNBT, and MNBNT). We have tried the experiment of adding more features to extend the maximum entropy to examine how well the performance will be by having more than one tag instead individual tag of ambiguous pattern. The few of ambiguous rules in the transfer file that we have learned and number of few ambiguous patterns in the corpus could be the causes of poor performance.

Furthermore, we have performed another experiment which is increasing the size of ambiguous words in the training corpus. Figure 4.9 shows the result based on several rate for ambiguous words in the training corpus. As it is obviously, the error rate was same with all the evaluation metrics for all sizes that means we did not achieve any gain by increasing the size of ambiguous words for trained monolingual corpus. As well as, we have recorded difference with the different coverage of the context rules for translation, which is depicted in Figure 4.8, the result could not enhance by increasing the size of the ambiguous words. Here for both experiments the reasons could be the same of stability of the result with increasing the number of ambiguous words in a training corpus, and these reasons are the few of ambiguous patterns and the few applied ambiguous rules in a structural transfer file.

In the meantime, Figure 4.6 shows the BLEU result which is automatic evaluation scores for measuring the quality of translations for the MT systems (weighted, TLM, and unweighted systems). In the first place, the unweighted system exhibit stable result by different size of training corpus because the unweighted system does not have training choice, we just evaluate it frequently with same test data, and having consistent figure with all baseline systems. The unweighted system record poor performance Compared to other systems (weighted and TLM). Also, The TLM, shows stable result with different size of training corpus, but it outperforms both the weighted system with size of corpus less than 10000 sentences in the training corpus. Lastly, the performance of the weighted system has enhanced gradually by increasing the size of training corpus. The weighted system records peak in performance with size of data more than 10000 sentences in the training corpus. Afterward, we have measured the performance of the weighted system compared with baseline system (unweighted and TLM) by utilizing another two metrics which are WER and PER as it is presented in the Figure 4.7. Generally, the weighted system outperforms the TLM system with different size of training corpus. As it is depicted in Figure 4.7 the error rate of both systems decreased by increasing the number of sentences in the training corpus. Notice that, we did not put result of the unweighted system here because it will not show any difference from its result with BELU metric. Furthermore, this is a much better enhancement in the performance and the confidence intervals overlap for all the metrics in the weighted system. Finally, to decide whether the improvement is statistically significant, we execute pair-bootstrap resampling for both measures amid the systems. We have compared all the systems against the unweighted system. The improvement in performance is statistically significant (p= 0.75).

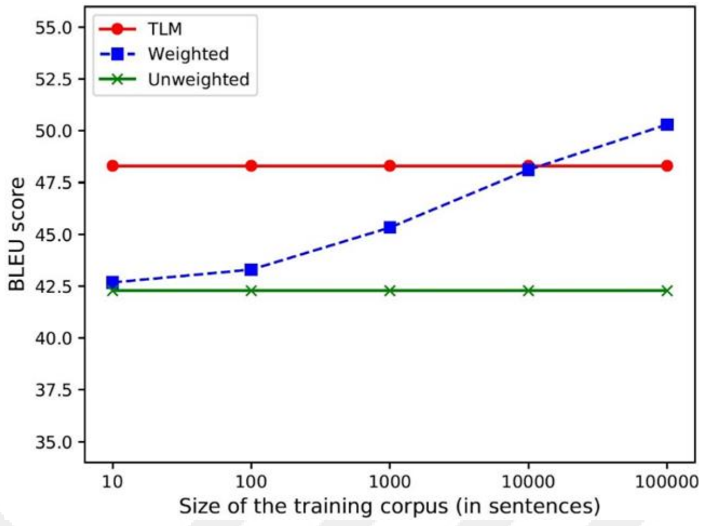


Figure 4.6: Translation quality measured for the baseline RBMT system (unweighted), target language model (TLM), and the weighted approach described in section 4.1 for the Kazakh–Turkish (for different size of corpus)

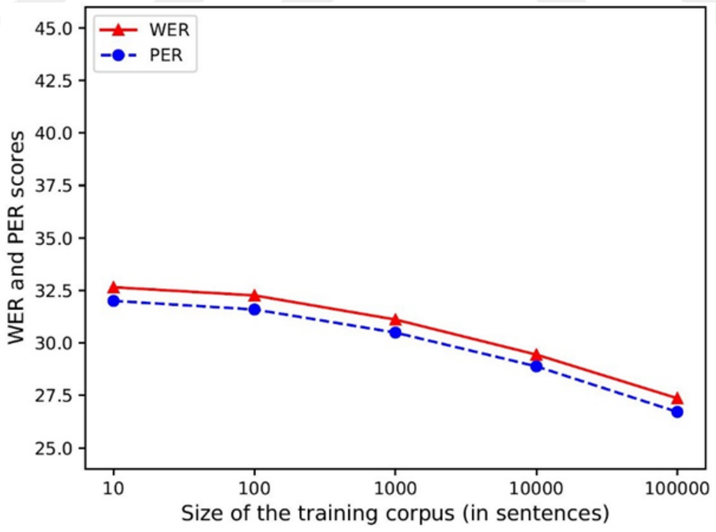


Figure 4.7: The graph depict how the error rate is diminish by adding more data into training corpus. The two horizontal lines are the weighted system and the target-language model best system

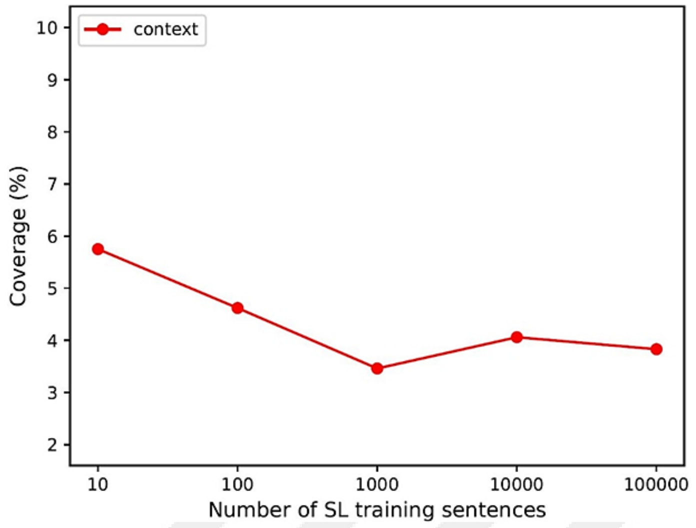


Figure 4.8: The graph display the coverage of the context rules for translation and record difference as increasing the amount of the training data

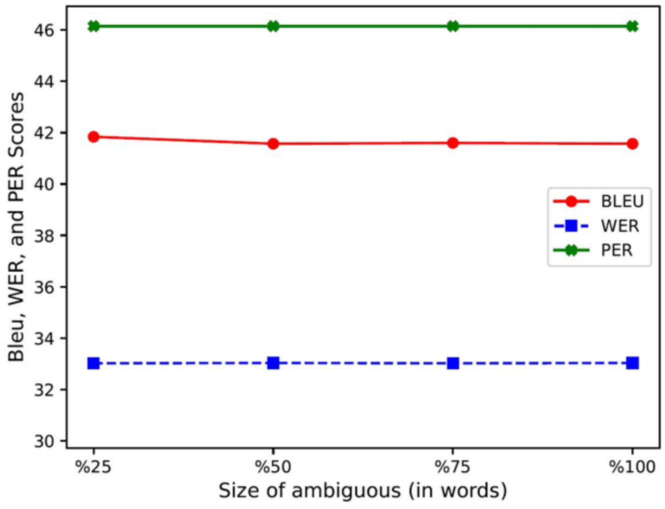


Figure 4.9: The graph depicts a consistently of the error rate, despite increasing the number of ambiguous words in a training data

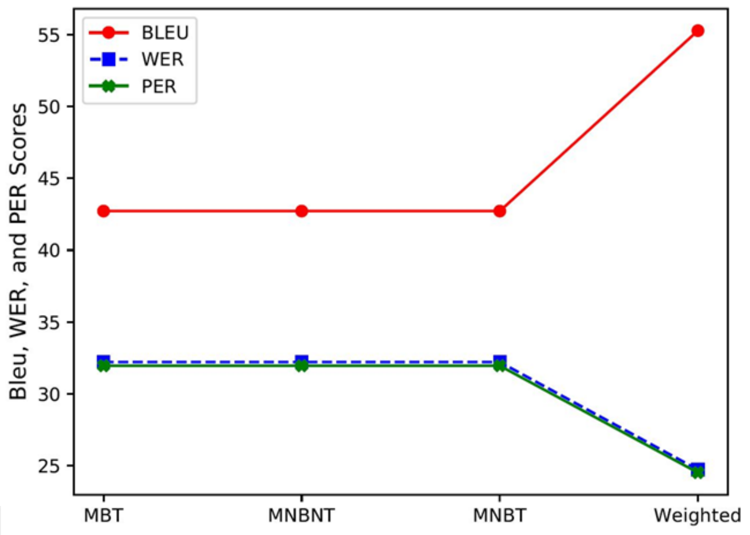


Figure 4.10: The graph shows comparison of the error rate between several models with different features, MBT, MNBT, MNBNT, and our weighted model (with unknown words and without tags) as more as ambiguous words is used for training. There is no difference in result among systems except weighted

5. SUPPORT VECTOR MACHINES

Support vector machines (SVMs) are a supervised learning methods, are first released by [103], [104]. They are among the state-of-the-art classification techniques [97]. SVMs consider substantial tools in dealing with large data classification and regression. SVM is a popular classifier and exhibits good performance on a several of different applications, including machine vision, pattern recognition and text categorization because they are robust when dealing with noise and sparse features [105], [106].

Basically, the principle of the SVM is determining linear separators (hyperplane) to divide the search space which can classify the diverse classes [107]. At the same time, the SVM minimizes the empirical classification error and maximizes the geometric margin. Therefore, the SVM named as Maximum Margin Classifiers. The SVM maps input feature vector to space (higher dimensional space) in which a maximal separating margin constructed [108], [109].

In other words, by giving labeled data it is performing classification by finding the hyperplane (decision boundary) that maximizes the distance margin between the support vectors of two classes. Support vectors represent the training data which is closest ones into decision boundary. Thus, on the ground of that hyperplane (decision boundary), the two hyperplanes are constructed that separate the data.

In two dimensional space this decision boundary is a line dividing a plane in two parts where in each class lay in either side. The margin is the interval from the support vectors to the hyperplane as depicted in Figure 5.1. In the Figure 5.1 the data made up of two categories, the SVM classify the classes $\boldsymbol{y} \in$ **{+1 , -1}** with hyperplane. In order to decrease the error rate of incorrectly linearly separating categories, the two categories on the sides should be negative and positive, plus the margin has to be quite large. In equation 5.1 which is a function of linearly separable of training data ***X.***

$$f(x; w, b) = \sum_{i=1}^{d} w_i x_i + b = w.x + b \tag{5.1}$$

$$sgn\big(f(x^i; w, b)\big) = y^i \ \forall \leq i \leq N \tag{5.2}$$

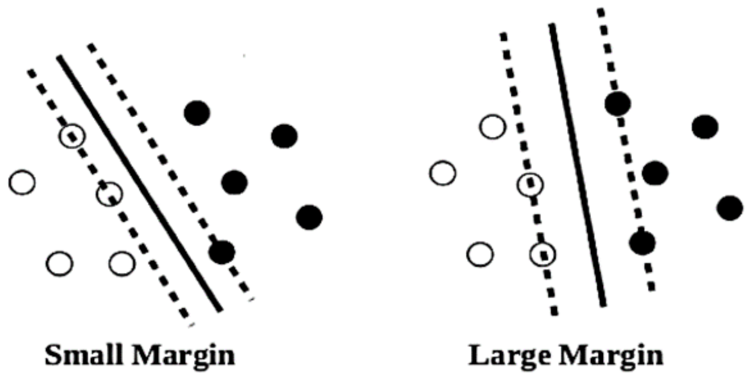


Figure 5.1: SVM where dot lines represent decision boundary, the data on dot lines are support vectors, the solid line is hyperplane, and the black, white circles represent negative and positive samples [112]

The classification come true employing the following discernment function 5.2, the two unknown variable are ***b, w*** bias and weight respectively that model will learn them during training. Where. $x^{(i)}$ and $y^{(i)}$ are known variables, where x is the context (or features which represent [n_samples, n_features]), $\mathbf{x^{(i)}}$, $\boldsymbol{y^{(i)}}$ $\boldsymbol{i} = \mathbf{1, 2, 3, ...}\boldsymbol{N}$ $\boldsymbol{y^{(i)}} \in \{+1 , -1\}$ and $\boldsymbol{y^{(i)}}$ equal to target/class.

On the other hand, when training data is not linearly separable, because the inner region of the margin has involved with some samples of training data. Therefore, the linearity of hyperplane will be changed into non-linearity that is by implementing either support vector classifier (SVC) or Nu-Support Vector Classification (NuSVC) or a linear support vector classifier (LinearSVC). Those approaches are capable of carrying out multi-class classification process on a training dataset. In this manner, the training data is classified into more than two classes, because the classes $\boldsymbol{y} \in \{+1 , -1\}$ could not be linearly separable.

The SVM has recorded influential certainty in solving both linear and nonlinear problems by maximizing the margin between classes [110]. Even though the SVM was originally formed for only a binary classifier problem that means it particularly classifies two class at a time, the advanced SVM mechanisms are able to compose a multi-class SVM by breaking up problem into several assorted binary problems [111]. However, in order to classify the data into multiple classes, i.e., more than two, SVM has to train two or more binary classifiers by grouping multiple classes into two classes. Such a strategies designed for these missions are “One-Versus-One” (OVO) scheme which is derived of the SVC and “One-Versus-Rest” (OVR) scheme, it is case of Linear SVC, both of them designed to handle the multiclass problem. The OVO and OVR methods are our focus in this research in which two pairs of classes are selected at a time and a binary classifier trained for them, these approaches have explained in more details at (section 5.2) and (section 5.4) jointly.

Additionally, the SVM consider a novel approach to solve various natural language processing issues. Researchers assume that the SVM is more capable to do elimination of unnecessary features and picking up the most appropriate features than other machine learning approaches [2]. This property can be achieved with involving a function by the SVM which eliminate all samples except the samples represent as support vectors. The precision is then improbable to decrease even there are many of unnecessary features. For example, murata et al. [112] have compared the performance of several machine learning algorithms include the SVM for Japanese-English translation of tense, aspect, and modality. They obtained the most accurate result by applying the SVM than applying other machine learning algorithms. The SVM in their work has been integrated with pair-wise method to perform classification for non-linearity data. In pair-wise method, data contents of N categories, it can be classified into pairs with (N (N-1)/2 pairs). Therefor the best category is detected by two categories classifier which obtained by N (N-2)/2 pairs. The pair-wise method is representing the OVO method that we implement it in this work

5.1 MULTICLASS SUPPORT VECTOR MACHINE

SVMs strained to acquire structural risk minimization (SRM) principle because they are based on variational-calculus and this utilize convex optimization with unique optimum solution [113]. The multiclass classification problem is essential continuation of binary classification problem that can be formulated as follows. Given a training set of multiclass classification problem:

$$T = \{(x_1, y_1), \dots \dots \dots \dots, (x_1, y_1)\} \tag{5.3}$$

Where, $\boldsymbol{x_i \in R^n, y_i \in y = \{1, ..., M\}, . = 1, ..., l}$. The class $\boldsymbol{y}$ for every new input $\boldsymbol{x}$ is assigning by a decision function $\boldsymbol{y = f\ \{x\} \in R^n}$.

When the problem of multiclass classification as presented above, there will be separations (hyperplanes) to classify the $\boldsymbol{R^n}$ space into $\boldsymbol{M}$ regions corresponding to the training set and one of these hyperplanes which maximize the margin is accredit an optimal separating hyperplane. In this case, the multiclass classification is pretended as “transformation to binary” strategy. Thus, the multiclass classification problem has to be reduced to multiple binary classification problems.

Furthermore, as mentioned previously the OVO and the OVR schemes are two common multiclass classification strategies (over m classes) depend on creating several binary classifiers. It could be possible to make the full separation at once, but this not consider cheap in term of computation and its performance not well in practice, therefore supplying the binary classifiers are well-tuned. The OVO scheme could be better in case of applying such a kernel algorithm which not perform scaling with n_samples. As each one is learning issue by just include a small subset of the data whereas, in the OVR scheme, the complete dataset is used n_classes times. In addition, the OVR scheme utilize a limited binary classifier and the training cost is definite with number of classes but it is criticized for no constrained on the generalization error [111] and also asymmetric approach can be used to solve potentially asymmetric problems [114]. On the other hand, the OVO scheme is easy to train because of each binary classifier of two class will be resolved by one classifier, however the computation cost is maxi because the number of binary classifier grow as $M * (M - 1)/2$.

In this chapter, firstly the multiclass classification problem is handling according to the OVO scheme by utilizing voting scheme based on combining many binary classification decision functions. The approach has developed based on decomposing the multi-class problem into multiple binary problems which also named as problem transformation techniques. The proprieties of the OVO scheme which have been mentioned previously make us think that the OVO scheme is more appropriate to our problem than the OVR scheme. However, to prove this indication we have implemented the OVR scheme too, their result discussed in the section 5.4.2.

5.2 ONE-VERSUS-ONE

SVC execute OVO technique [115] to solve the multiclass classification problem. The SVC finds the linear hyperplane which separates the classes with maximum margin. Suppose the OVO scheme consists of multi linear single class SVMs. Thus, the training has to be perform only on the instances relevant to the two OVO classes interest ***a*** and ***b*** as indicated in equation 5.4. The subset of data giving for training as below:

$$D_{a,b} = f(x) = \{(x_i, y_i) | x_i \epsilon \mathrm{R}^p, y_i = \left\{\begin{pmatrix} 1 \text{ if cllss } (x_i) = a \\ -1 \text{ if cllss } (y_i) = b \end{pmatrix}\right\}_{\boldsymbol{i=1}}^{\boldsymbol{n}} \tag{5.4}$$

For instance, *i*, the feature vector represents as $\boldsymbol{x_i \epsilon \mathrm{R}^p}$. The number of examples which belong to either class ***a*** or class ***b*** can calculate the number of training ***n*** from the ***N*** training instances available for all classes.

$$min \frac{1}{2} \|w\| + c \sum_{i=1}^{n} \epsilon_i \tag{5.5}$$

$$s.t: yi(wT\ xi + b) \geq 1 - \epsilon_i \ and\ \epsilon_i \geq o \forall i\ \epsilon \{1 \dots \dots n\} \tag{5.6}$$

In the equations 5.5 and 5.6 above the ***w*** and ***b*** represent the weight and bias parameters for linear model. And ***c*** denotes to parameter which controls how the model fits input data and should be tuned earlier.

If ***M*** is the number of classes, then for different ***i, j*** there will be ***M* (*M*-1)/2** decision functions and each one trains data from two classes. Generally, the decisions functions try to predict the probable class y for every new input *x* based on which classes gets highest number of votes; a vote for a specific class is defined as a decision function for assigning the new input ***x*** into that class. Note that, when there are two or more than two classes with the same number of votes, then the input *x* is unclassified in this approach. An example of the OVO scheme, assume we have a 3 classes problem, with classes y_1, y_2, y_3. The samples will be $x_1, x_2, ..., .n$ and the classifiers have to be $f1, f_2, ..., f_n$. Nevertheless, speculate your training data is **{**{x_1, y_1}, {x_2, y_1}, {x_3, y_2}, {x_4, y_1}, {x_5, y_2}, {x_6, y_3}, {x_7, y_3}**}**. Thus, f_1: trained with the subset **{**{x_1, y_1}, {x_2, y_1}, {x_3, y_2}, {y_4, y_1}, {x_5, y_2}**}**, for classes y_1 and y_2, f_2: trained with the subset **{**{x_3, y_2}, {x_5, y_2}, {x_6, y_3}, {x_7, y_3}**}**, for classes y_2 and y_3. f_3: trained with the subset **{**{x_1, y_1}, {x_2, y_1}, {x_4, y_1}, {$x_6, y3$}, {$x_7, y3$}**}**, for classes y_1 and y_3.

```
Algorithm 1 One-vs-One algorithm
1:   For each i, j that $1 \le i \le j \le K$,
2:   (a)Take the original $Sw = \{(xn, yn, wn)\}_{n}^{N} = 1$ and construct a binary training set
     $S_b^{(i,j)} = \{(xn, yn, wn) : yn = i \text{ or } j\}$
3:   (b) Use a weighted binary classification algorithm $Ab$ on $S_b^{(i,j)}$ to get a binary classifier $g_b^{(i,j)}$
4:   Return $\hat{g} = \underset{1 \le l \le M}{argmax} \sum_{i<j} \left[ g_b^{(i,j)}(x) = l \right]$
```

The algorithm 1 demonstrates a weighted version of the OVO scheme which decomposes the multiclass classification task into $M\ (M-1)$ binary classification subtasks. In consequence of the $O\ (M^2)$ increases in the number of the subtasks, the OVO scheme is generally more suited when *M* is not too large [116]. Here each binary classification subtask consists of comparing samples from two **(*i, j*)** classes only. For each $g_b^{(i,j)}$ *b* aim to predicate whether ***X*** ”favor” class ***i*** or class ***j***, and $\hat{g}$ predicts with the preference votes collected from those $\hat{g}_b$.

5.2.1 Kernels in Multiclass Support Vector Machines

Kernel functions set up the aspects of the SVM model and level of nonlinearity. So far finding an appropriate separating hyperplanes are constrained task for separating input data that drawn from

R^n. For data are linearly separable can easily separating into two classes by applying the function ***f* (*x*; *w*, *b*)**.

Nevertheless, in linguistic tasks, we frequently deal with data that is not linearly separable, or the input data comes from discrete space (such as the space of all words, or all strings, or trees or graphs, etc.) instead of coming **R^n** [117]. In case the data is not linearly separable, but it is drawn form **R^n**. Then, we would be capable to solve this problem by mapping the input data form **R** to R*n* by feature function **Φ (*x*) = (*x* ,*x^2*)** Where **R*n*** is input space (It implies to the space in which the *.s* is mapping) and *x* is a feature vector. Figure 5.2 is an illustration of how one-dimensional and non-linear dataset has been separated in higher dimensional feature space (It is referring to the space from which the x_s are classified). The dataset in the left side of the Figure 5.2 is not linearly separable. After we map the dateset form the one-dimensional space into the two-dimensional space, we get the dataset displayed in right side of Figure 5.2. Here, the input data can be linearly separable, but the computational complexity will explode exponentially.

Then, it would be capable to use one of 'kernel trick' to solve the problem of non-linearity by mapping input space **$I \in IR^n$** into a, generally higher dimensional, feature space ***F*** where a hyperplan can separate the classes. Therefore, when training data are not linearly separable, in such case the training algorithm will rely on the data over inner products (usually the dot product) in the feature space ***F***, because datapoints in the training issue will come out as inner products, e.g. in the assemble φ **(x_i). φ (x_j)**. The purpose of the kernel trick is to calculate the kernel function like that ***K* (*xi,xj*)= φ (x_i) . φ (x_j)**, because for the training the only ***K*** would be required, the **φ** has to be unknown.

In addition, the most commonly used kernels which also have been used in this work are linear kernel, radial basis function (RBF), and sigmoid kernel.

- **Radial basis function** the kernel acquired from the process of neural network society RBF, as well as known as the "squared exponential" kernel. This kernel usually has infinite feature space. The RBF has one hyperparameter denoted as γ. The parameter γ manages how close observations have to be to contribute to the classification decision. Accordingly, the ***K* (x_i,x_j)** used in training algorithm, the SVM approach can easily generate support vectors that placed in infinite dimensional space. The RBF kernel is more suitable than linear kernel and it consider a default choice. Classification with RBF kernel can be achieved by equation 5.7.

$$k(x_i, x_j) = (-\gamma \|x_i - x_j\|^2)\,) \tag{5.7}$$

In addition, while training input data with the RBF kernel, the two most important parameters have to be evaluated well: ***C*** and γ. The parameter ***C*** which is used with all types of kernel unction, exchange misclassification of training examples against simplicity of the decision surface. Thus, the small ***C*** creates the decision surface smooth, and a large ***C*** intends at classifying all training examples correctly. γ describe how much influence a single training example has. The larger gamma is the adjacent other examples have to be affected.

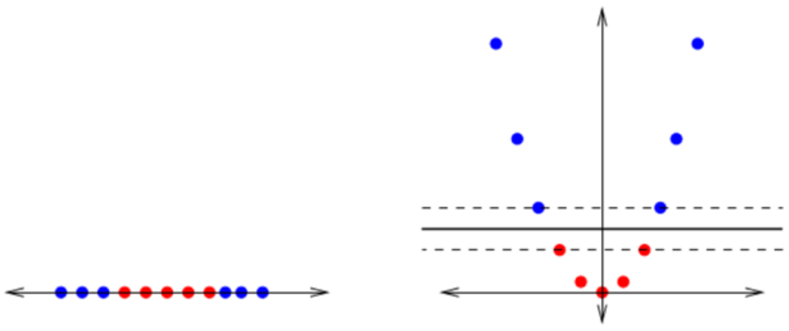


Figure 5.2: Mapping Non-linearly separable data on the left in one dimension input space into linear separable data in tow dimensions feature space on the right [117]

- **Sigmoid kernel** is also known as Hyperbolic Tangent kernel, it is first released in [49], and successfully applied in some practical cases [118]. This kernel quite popular for the SVM, because it comes from neural network field. The SVM approach employs sigmoid kernel function which is equivalent to a two-layer, perceptron neural network. The Sigmoid kernel does not perform well as another kernels due to its fundamental lacks for demands of a valid kernel. Parameters ***k*** and ***σ*** are adjustable in the sigmoid kernel as referred by equation 5.8. The slope ***k*** and the intercept constant ***σ*** have to be choosing properly to acquire good classification efficiency [119].

$$k(x_i, x_j) = \tanh(kx_j^T - \sigma) \quad (5.8)$$

- **Linear kernel *K* (*x*, *y*) = $\boldsymbol{x^T y + c}$** is a simplest kernel function, it depends on the penalty parameter $\boldsymbol{c}$, for that the reason, it is reducing the support vectors, training error and classification error by increasing the parameter $\boldsymbol{c}$. However, the linear kernel is not appropriate for huge datasets.

The linear kernel is a straightforward kernel over $\mathbf{R}^n$. For this kernel the feature space is accurately the same as the input space. One of the benefit of the linear kernel, its training of a SVM with a Linear Kernel is quicker than with any other Kernel, therefore, this kernel very convenient, when the number of features is larger than the number of observations (or support vectors).

The dateset is demonstrated in the Table 5.1 is an example of sample data format which is using as input data to the SVM's kernels. Here n_samples (class) is the target, fractional count are the sample weight and pattern words are the features. In this example, we implement the linear kernel on the input data. The reason of applying linear kernel is that our input data is linearly separable. The data is constructed form four combinations, because there are two ambiguous patterns. We will have one classifier model for each ambiguity. The ambiguous patterns are *секіргенін*,'splash', *sıçramak* and *көру*, 'seeing', *görmek* respectively.

Table 5.1: The input data format of linear kernel, where rule/class is the target, fractional count is the sample weight and pattern words are the features.

n-samples	sample-weights	n-features
0	0.22535	[Акуланың су бетіне секіргенін көру]
1	0.31701	[Акуланың су бетіне секіргенін көру]
2	0.14693	[Акуланың су бетіне секіргенін көру]
3	0.31071	[Акуланың су бетіне секіргенін көру]

Table 5.2: The input to the first classifier model will be word секіргенін, ‘splash’, sıçradığını or cıçrayarak, 1 stand for class/rule 1 and 0 stand for class/rule 2, the output will be 0, we choose the class 0 for the output sıçradığını.

n-samples	sample-weights	n-features
0	0.31071	[Акуланың су бетіне секіргенін көру]
1	0.31701	[Акуланың су бетіне секіргенін көру]

The two classifier models are illustrated in the Table 5.2, and Table 5.3 respectively. In each of the classifier model we have two classes. Here 0 stands for class1 and 1 stands for class 2. For the first model, the input will just be the word *секіргенін*, ’splash’ and the output will be 0 which refer into class 2, then, we will choose the translation *Sıçradığını*, ’splash’. And for the second class the input will be *көру*, ‘seeing’, and the output will be 1 that is refer into class1, thus, we choose *görmek*, ’seeing’. The translation of the sentence *Акуланың су бетіне секіргенін көру* will be *Köpek balığının su yüzüne sıçradığını görmek* ‘Seeing the shark splashing in the water’ which is a second combination as clarified in the Table 5.4.

Table 5.3: The input to the second classifier model will be word *көру*, ‘seeing’, *görmek* or *görme, a* stand for class/rule 1 and 0 stand for class/rule 2, the output will be 1, we choose the class 1 for the output *görmek*.

n-samples	sample-weights	n-features
0	0.31071	[Акуланың су бетіне секіргенін көру]
1	0.14693	[Акуланың су бетіне секіргенін көру]

Table 5.4: The second combination out of four combinations for sentence *Акуланың су бетіне секіргенін көру*, ‘Seeing the shark splashing in the water’ *Köpek balığının su yüzüne sıçradığını görmek* will win by applying the linear kernel.

rules	target
[rule-45][rule-12]	[Köpek balığının su üzüne sıçrayarak görmek]
[rule-46][rule-12]	**[Köpek balığının su yüzüne sıçradığını görmek]**
[rule-45][rule-12]	[Köpek balığının su yüzüne sıçrayarak görme]
[rule-45][rule-12]	[Köpek balığının su yüzüne sıçrayarak görmek]

5.3 RULE APPLICATION FOR SUPPORT VECTOR CLASSIFIER

The procedure of applying rules that we are following here is a same as the previous chapter just the applied weights learning algorithm is different. Note that, we use the same algorithm in the previous chapter to apply transfer rules on the input sentences for getting all possible combinations for ambiguous patterns. First, we apply the rest of the apertium tools biltrans and lexctor on the input sentence to get the lexctor which is a string of tokens (words) each with its translations and part of speech tags. Second, we would split these strings into source and target tokens along with their tags. Third, we will match these tags with categories from the transfer file as these matches will help us to match the tokens to the transfer rules.

Forth, transfer rule’s function is matching categories with pattern items of lexical items and applying a sequence of actions. There could however, be more than one potential sequence of actions for each source language pattern. In this case the rules apply to the patterns of input sentences and then matched rules to the input sentences will detect as the active features. If more than one rule is applied to the same pattern that means, there is ambiguity with that pattern. And if many patterns have ambiguities that makes the whole sentence has much more ambiguity, as all the possible combinations are equal the multiplication of each number of ambiguous rules of each pattern.

The dilemma concerning these ambiguous rules can be addressed by computing scores for sentence variants with a probabilistic language model and detecting the presumed rule by relying on the policy of the OVO classifier scheme.

Through the training process, every feature is assigned a weight λ^s and by using one of the OVO possible classifier kernels, the weights of ambiguous rules will be learned for each SL pattern. When there are just two ambiguous rules apply to one SL pattern, then, the OVO classifier scheme will be sole one classifier model (one binary classifier model). But, if there are more than two ambiguous rules, the one classifier model that does binary classification will not be able to perform classification for input data. Here, the SVM with kernels will use OVO to perform classification process. Because the OVO model deals with multiclass classification problem, it considers each binary pair of classes and trains classifier on subset of data containing those classes. For a coverage of M different ambiguity combination there will be $M * (M-1)/2$ classes/models. Thus, the training process will be repeated $M * (M-1)/2$ for each pattern. For example, if we have 15 classes, we need to create **(15*14)/ 2 = 105** binary classifiers. The illustration of whole training process is given in Figure. 5.3. During the classification period each classifier predicts one class. In the OVO model, the voting strategy are applied, the input sample have to be tested with all models and record how many times a class is preferred with respect the others and which class having the majority of votes wins. And if two classes have same number of votes, then will be ignored. This classifier is then integrated into the translation model. In Figure 5.4 we illustrate the structure of the prediction process with different kernels. The prediction procedure and the training process are same for all the kernels.

5.4 LINEAR SUPPORT VECTOR CLASSIFIER FOR TRANSFER-RULE SELECTION

LinearSVC is an implementation of SVM classifier for the case of a linear kernel. It should be indicated that the LinearSVC accepts keyword ‘kernel’ with parameter kernel=‘linear’, because it considers to be linear. The input data format of the LinearSVC has to be as two arrays which is default to “fit (self, X, y, sample_weight=None)”: an array of ‘X’ of size [n_samples, n_features], n_features represent pattern words, ‘y’ is [n_samples, n_classes] that the LinearSVC function going to be predicate, fractional count is the sample weight. However, when input data points are belonging into ‘M’ different classes this consider problem of multiclass classification. Therefore, the multiclass support will be handled according to the OVR scheme also named as One-Versus-All (OVA). The OVR strategies include training one classifier for every class, with the samples of that class as positives samples against other classes with negatives samples, and that’s because the SVM only deals with binary classification problems. The OVR scheme could obtain real-valued confidence score for its decision by using the base of classifier, individual class labels alone can lead to ambiguities, where multiple classes are predicated for an individual sample.

The OVR considers one of the straightforward multiclass classification algorithms constructed on top of real-valued binary classifiers to train ‘M’ different binary classifiers, that is for distinguishing the samples in a single class from the samples in all other classes or models, each classifier will be trained separately. Whenever it is requested to predicate a new sample data, we run the ‘M’ classifiers, all model classes take unseen sample and give either a probability score denoting it’s belonging to that model class, or a class label. After that we classify the new sample with highest probability class model. Correspondingly, the OVR classifier are using as logistic regression by turning the problem into binary classification problem. Therefore, the samples of one class will be turned into positive examples, and the rest of classes into negatives. This is how OVA working throughout this thesis. It is conceptually simple, and has been independently invented numerous times by various researchers.

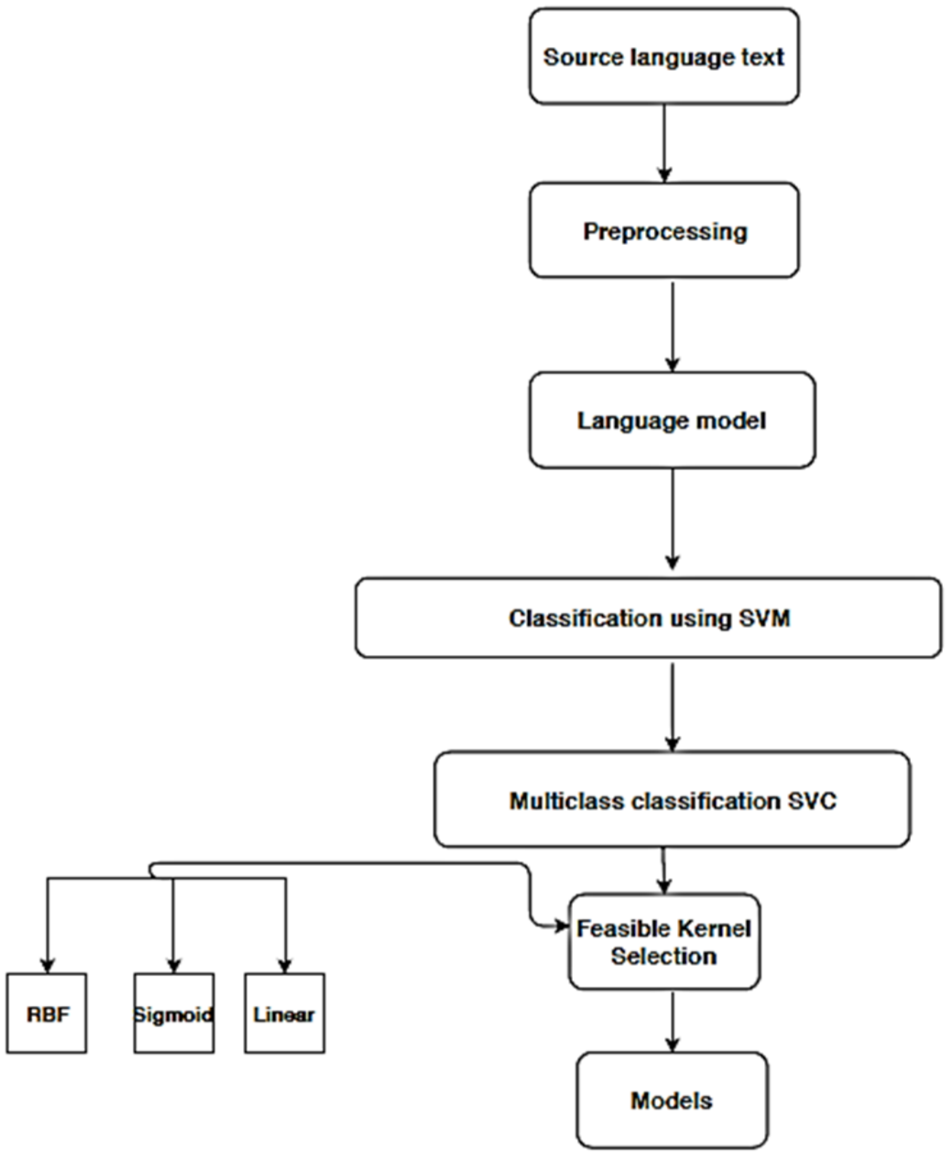


Figure 5.3: Flow of support vector classifier training process

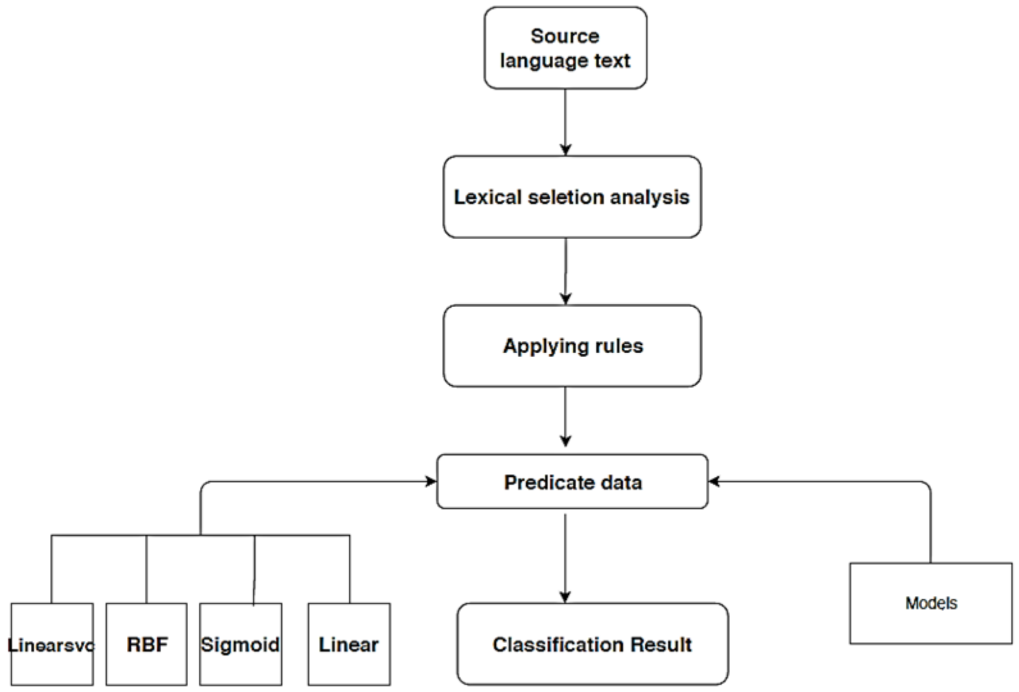


Figure 5.4: Flow of support vector classifier predicting process

The OVR scheme can be consists as follows: we frame $\boldsymbol{m}$ classifiers $\boldsymbol{f_i}$ **for** $\mathbf{1} \leq \boldsymbol{i} \leq \boldsymbol{k}$, where class $\boldsymbol{i}$ is zero and all other classes are one for $\boldsymbol{f_i}$. After that, we pick up the category as the value *i* which has maximal value of $\boldsymbol{f_i}\ (\boldsymbol{x})$ for a datapoint *x* as described in equation 5.4. Besides its computational efficiency (merely n_classes classifiers are desired), the benefit of this scheme that can be interpretable. The knowledge about the class may be get by checking its matching classifier, and that happened because per class assigned by one classifier. This consider the most regularly applied method in case of multiclass classification and is a fair default choice. Below in a Table 5.5 we indicate an example of input data structure that LinearSVC will accept.

$$\check{y} = \underset{k \epsilon \{1, \ldots, m\}}{argmax}\ f_m(x) \tag{5.9}$$

In our problem to select the most probable transfer-rule among several ambiguous rules (rules combinations) that match one pattern, we need a classifier for sorting out input patterns into two or more categories based on the number of classifier we have. The OVR method is a good fit scheme for this problem. Therefore, we may shift this problem into multi binary classification problems (i.e. where we predict only 'y' ϵ 0,1). In the OVR scheme, we will have 'n' class models, where ‘n’ is the number of class (rules combinations), and it’s equal to ‘3’ in the example for Spanish-English pair data which is putted in a Table. 5.6. The whole training process of linearSVC by using the OVR is illustrated in Figure 5.5.

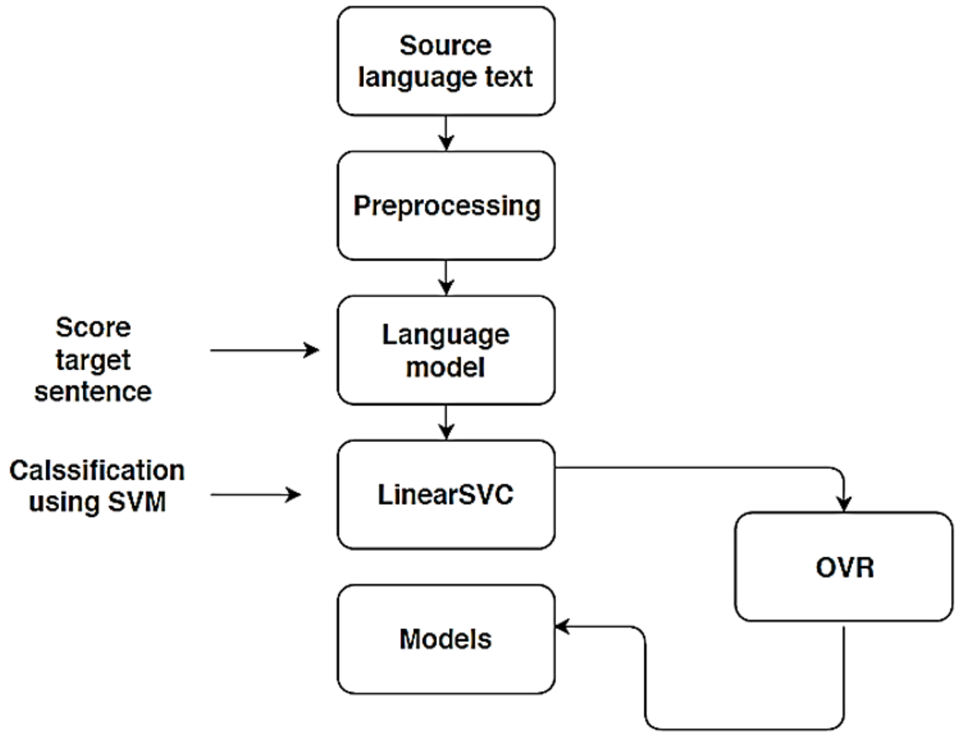


Figure 5.5: Flow of linear support vector classifier process

The sample of dataset is illustrated in the Table 5.6 is a three combinations of ambiguous rules which consist of two longest rules and two shortest rule, where n_samples (class or rule) is the target, fractional count are the sample weight and pattern words are the features. First and third combinations are made of different longest rules, and second combination is composing of two shortest rules. In SL sentence *Encuentro el pastel muy bueno*, ‘I find **the cake very good**’, the ambiguous patterns (features) are ‘el pastel muy bueno’ match 4 ambiguous rules. The pattern words in Spanish which form of ‘determiner noun adverb adjective’ has three forms in English as ‘determiner adverb adjective noun’, ‘adverb adjective noun’, and two short forms as ‘determiner noun’, ‘adverb adjective’. Here, the SL sentence Encuentro el pastel muy bueno with the form ‘determiner adverb adjective noun’ translated as ‘I **find the very good cake’**, with the form ‘adverb adjective noun’ like ‘**I find the very good cake**’ and with the two short forms as ‘**I find the cake very good**’

Thus, the OVA model will contain three different models per each class that is for training each class against the two other classes, each one marks the samples of that class as positive and other samples as negative. Table 5.6 is an example of one of three models that can be produced from the sample dataset in the Table 5.5, here **+1** stands for class **1** and **-1** stands for other classes which denote as class 2. In a predication process, each of the three models takes unseen sample and gives either a probability score denoting its belonging to that model class, or a class label. In a Table 5.7, the three classes is indicated with the applied rules. First, we do prediction using the first model, and it should give us a probability for being classified as +1 (rule1+rule2), or a class label either **+1** or **-1**. Second, we do prediction using the second model, and it should give us a probability for being classified as **+1** (rule1+rule3+rule4), or a class label either **+1** or **-1**. Then, we do prediction using the third model, and it should give us a probability for being classified as +1 (rule1+rule5+rule6), or a class label either **+1** or **-1**. Finally, we classify this sample with the highest probability class, or the class predicted as **+1** by any model. Here, the sample will be classified with class **2** of model **1**, because it is a highest probability class.

5.4.1 Experiments

As we are learning the ambiguous rules and weights with LinearSVC using the OVA scheme or the OVO scheme kernels by supervised training of source language corpus, we have operated the same evaluation metrics used in the previous chapters. Nevertheless, for calculating the weights of the binary feature, we utilize the implementation of generalized repetitive scaling available in the sklearn. This implementation is for linearSVC and the OVO kernels.

Table 5.5: The input data format of LinearSVC for OVA model, where rule/class is the target, fractional count is the sample weight and pattern words are the features.

n-samples	sample-weights	n-features
0	0.22535	[Encuentro el pastel muy bueno]
1	0.42712	[Encuentro el pastel muy bueno]
2	0.34753	[[Encuentro el pastel muy bueno]

Table 5.6: Sample One of Three sub-models of the OVR model, here +1 stand for class 1 and -1 stands for other classes which they consider as class 2. Where rule/class is the target, fractional count is the sample weight and pattern words are the features.

n-samples	sample-weights	n-features
-1	0.22535	[Encuentro el pastel muy bueno]
+1	0.42712	[Encuentro el pastel muy bueno]
-1	0.34753	[[Encuentro el pastel muy bueno]

Table 5.7: Sample of data when there are two ambiguous words in one sentences. and four combination generated by applying four ambiguous transfer rules.

Rules/Classes	Target
[rule-1][rule-2]	[I find the very good cake]
[rule-1]][rule-3] [rule-4]	[I find the cake very good]
[rule-1]][rule-5] [rule-6]	[I find the very good cake]

5.4.2 Results and Discussions

In this section, we first compare the various of the OVO kernel's schemes with the LinearSVC scheme. All kernels algorithms of the OVO-type obtain a multiclass $\hat{g}$ by calling a weighted binary classification algorithm $\boldsymbol{A_b}$ for $\boldsymbol{M}$ **(*M*-1)/ 2** times. While in the LinearSVC, we build $\boldsymbol{m}$ classifiers $\boldsymbol{f_i}$ **for** $\boldsymbol{1 \leq \ i \leq m}$, where class $\boldsymbol{i}$ is positive and all other classes are negative for $\boldsymbol{f_i}$. For prediction, the OVO kernels functions requires gathering votes from $\boldsymbol{M}$ **(*M*-1)/ 2** form binary classifier. And the LinearSVC chooses the class or category as the value $\boldsymbol{i}$ which has maximal value of $\boldsymbol{f_i(x)}$ for a datapoint $\boldsymbol{x}$. We get the SVM for the OVO scheme with perceptron kernel [79] as *Ab* with all the observations and use LIBSVM [120] as SVM solver for our problem. Notice that a strong classification algorithm will be created by using perceptron kernel [121] and have ability to achieve a weighted binary classification [122]. In addition to that kernels, we have applied a linear kernel too which is known as fragile classifier. The LinearSVC is an implementation of SVM classifier for the case of a linear kernel and multiclass support handled according OVR scheme.

Moreover, we evaluate the result obtained by all LinearSVC and the OVO SVM's kernel functions through three different translation quality measurement metrics WER, PER, and BLEU. These evaluation metrics calculate the error rate of the text produced by the system compared to the postedited version of the same system as a reference.

In the Table 5.8, we have compared the result of the new system (using one of different kernels in multiclass SVM) with our weighted approach, the default (unweighted), randomly selected, and results are obtained by using the TL model online. At the same time, the results illustrated in the Table. 5.9 represent consequences of all the LinearSVC and the OVO kernel functions that have been applied through this work. For experimentation, the used dataset for the language pair (Kazakh, Turkish) are taken from Kazakh, and Turkish Wikipedia. The dataset which is utilized in the previous chapter is used for training and testing the SVM kernels in this chapter too. We have applied different kernels of the SVM in multiclass and their performance has been compared. As a result, the favorable kernel of this dataset is the RBF kernel.

In the Table. 5.8, the weighted system outperformed all the reference systems and different SVM kernels. The Weighted system recorded a baseline of WER **27.72**, PER **26.51**, and BLEU **53.28**. Nowadays, even though SVM classifier is one of the state-of-art approaches, but with our experiments show that the SVM classifiers methods could not overcome maximum entropy approach.

Table 5.8: Word error rate and Position-independent word error rate; OOV is the number of out-of-vocabulary (unknown) words. WER and PER scores with 75% confidence intervals for the reference systems on the test corpora. Note that the BLEU scores are computed against a postedited reference translation.

System	OOV (%)	WER (%)	PER (%)	BLEU
Weighted	0.28	27.72	26.51	53.28
Unweighted	0.28	32.14	31.88	42.28
LinearSVC	0.28	32.14	31.88	47.49
TLM	0.28	29.99	29.22	48.31
Random	0.28	30.91	30.00	49.73
Sigmoid	0.28	29.48	28.99	47.61
Linear	0.28	30.08	29.61	46.77
RBF	0.28	28.25	27.56	46.13

On the other hand, we achieve same experiments to compare the performance of the LinearSVC scheme and OVA kernel's schemes. As depicted in the Table 5.9, although results do not show a big difference, the RBF kernel's outcome surpass the LinearSVC and other two the OVO kernels Sigmoid and Linear. In the literature, several articles had recorded that the OVO scheme show better accomplishment than the OVA scheme [123]. Our result does not agree with this evidence which presented in all these studies, because the OVO scheme is ahead of the OVR scheme.

Table 5.9: Word error rate and Position-independent word error rate; OOV is the number of out-of-vocabulary (unknown) words. WER and PER scores with 75% confidence intervals for the reference systems on the test corpora. Note that the BLEU scores are computed against a postedited reference translation.

System	OOV (%)	WER (%)	PER (%)	BLEU
RBF	0.28	28.25	27.56	53.13
Sigmoid	0.28	29.48	28.99	47.61
Linear	0.28	30.08	29.61	46.77
LinearSVC	0.28	32.14	31.88	47.49

6. CONCLUSIONS

6.1 SUMMARY

In this thesis, the goal was improving the translation performance of shallow-transfer rule-based machine translation by developing a new module of selecting transfer-rule based on context. These will include three approaches as follows:

- Create Kazakh-Turkish machine translation system as presented in Chapter 3, in which the structural transfer rules has been formalized based on source language context, building a bilingual dictionary by analyzing parallel corpus of Kazakh-Turkish pair that has been created manually. During create the bilingual dictionary we use a lexical transfer module as declared in section A.2.4. As well as, we have also write some of lexical selection rules according to our needs to obtain a correct before applying the transfer rules on SL. A lexical selection deals with SL words when these words have more than one translation in TL, and try to choose the appropriate translation based on the context, in section A.2.5 we explain the lexical selection rules in deep.
- Learning weights of structural transfer rules with unsupervised learning approach, for this we have used one of statistical machine translation algorithm that is a maximum entropy approach. The whole work applying these algorithm as pretended in Chapter 4.
- Learning weights of structural transfer rules by applying another unsupervised learning algorithm which is presented in Chapter 5. The algorithm we have applied is a support vector classifier algorithm to learn weights of rules based on context.

Structural-transfer rules formalism and its implementation in finite state transducer based on transfer-selection module, its formalism was fairly straightforward after understanding the basic concepts of writing structural-transfer rules. These rules match fixed-length patterns, they do not involve any recursive rules, also without optionally at the level of words, that means it was just one. Though vocabulary coverage, the Kazakh-Turkish MT system has registered an ideal performance than competitive systems (Google and Yandex), the Kazakh-Turkish MT system would perform significantly better by expending the dictionaries with new lists of stems, besides providing bilingual correspondences. In addition, adding more transfer rules into structural transfer file by analyzing more text and finding grammatical and structural differences between Kazakh and Turkish. Thus, when the Kazakh-Turkish MT has quite enough of transfer rules, wealthy dictionaries, and all recent problems will be fixed, then, the Kazakh-Turkish MT system can be released. The released Kazakh-Turkish MT system will also boost the performance of the weighting MT system of applying learning algorithms on transfer rules.

Moreover, we have introduced a novel unsupervised learning approach of learning transfer rules from monolingual corpora. Since a parallel corpora are not always accessible for many less-resources language pairs, with our new approach it becomes possible to learn rules by using monolingual corpora. This method works like that: After we obtain all possible translations combinations of source sentences by applying ambiguous transfer rules for first ambiguous pattern separately from other ambiguous patterns which have been translated by applying LRLM method. After that, we have to score these rules with the n-gram language model for each of the ambiguous sentences for that pattern. Then, we normalize the received scores (fractional counts), and train the system with these normalized fractional counts beside source language patterns for generating the maximum entropy models which will use by Beam search algorithm to choose the best (highest weight) ambiguous rule that could be apply into input pattern of SL.

When more than one transfer rule could be applied to a given pattern, this is called an ambiguous rule. Apertium resolve this ambiguity by applying the left-to-right longest match (LRLM) method on transfer rules, and this not adequate with all the word/s that follow that pattern/s. To enhance this resolution, a new module was introduced to make these ambiguous rules are weighted for the word/s that follow the ambiguous pattern, and these performed by training a monolingual corpus with unsupervised learning algorithm to generate a maximum entropy models that are used to choose the best (highest weight) ambiguous rule to apply.

For a given input source sentence, we just generate all possible combinations for the first ambiguous pattern and choose left-right-longest match rule for other patterns in one sentence. We, then, get scores from the n-gram model for each of the ambiguous sentences for that pattern. Along with, these scores (or a fractional counts) are then written into some files, some files contain the scores of an ambiguous pattern and some other files contain non ambiguous pattern. These files are considered the datasets for a YASMET tool, which trains the TL maximum entropy models. Thus, in place of choosing the longest rules for first word, we sum up the weights of active rules for source language word the weights of the rules that are active. In meantime, having these models mean the new module of structural transfer rules are ready to use. Eventually, we perceive the translation with height sum of weights by applying the beam search algorithm. The beam search algorithm makes us capable to choose the best possible ambiguous rules to apply, hence having the best translation. The method shows a statistically substantial enhancement on simply choosing online the best association for each sentence accordant to the score returned by the target-language model.

Additionally, we have also applied another learning algorithm and we hoped for getting best result than applying the previous approach (unsupervised learning algorithms). For this we used support vector classifier algorithms because these algorithms are more fitted to our problem. As our problem is there could be more than one rule match same pattern of SL, then we will have multiple classes. Thus, we will have class for each ambiguous rule and by using support vector classifier algorithms we will be able to choose the appropriate class/rule has to apply to SL pattern for getting an adequate translation. Unfortunately, the performance of these algorithms was not much better than unsupervised algorithms.

We have implemented an efficient structural transfer model which is favorable for general purpose within Apertium created by Forcada et al. [124]. All of the software in this thesis is released as free/open-source software under the terms of the GNU General Public License, which ensures that the experiments are reproducible and allows other researchers to improve on them without having to reimplement the algorithms from scratch. The main idea of this work has presented by Bayatli et al.[125].

6.2 FUTURE WORK

Our imagination in light of the results obtained in the evaluation of the novel approaches that have been presented in this thesis is the limit only what will be the desirable future lines of research that could be discovered based on this research:

1. The all possible translations are obtained by using one of structural transfer method (coverage algorithm) of Kazakh-Turkish system. We used language model method on-line to score these target sentences and we have applied 6-gram model of surface form — as an initial experiment in both chapters 5 and 4. Though, its performance was not poor, trying other language models with more or less structure could improve the performance much better. It may be appealing to try different language method for getting more appropriate scores of target sentences.

2. The Kazakh-Turkish MT system which is presented in chapter 3 has not release yet. The system could be release after fixing all the errors which appear in analysis step. These errors cane be understood with special characters attached to the words in target sentence such as ('*', '#', '@'), each character assign to different problem, such as an analyzed word, untranslated lemma, and Unable to generate surface form from lexical unit respectively. The released system could positively affect its performance.

3. In this thesis we have only worked with one language pair which is Kazakh-Turkish pair, and both languages are morphologically rich languages, it would be worth if we apply it on more language pairs, that may help us more to appraise the performance of the system.

4. In this thesis we have applied two machine learning methods which are a Maximum entropy as unsupervised algorithm which is described in chapter 4 and as supervised algorithm we executed a support vector machine, it is presented in chapter 5, both algorithms used to learn weights of transfer rules. We have achieved almost close results with two methods. It will probably be better to try another state-of-art machine learning algorithm, markedly the neural network method which is highly preferable these days. By

trying Neural network method, we can more sure if the Neural network method could be preferred for less resource languages or not.

5. Since the coverage of the RBMT dictionaries have substantial effect on the performance of weighted system, the words that cannot be analyzed, therefore would not appear in the bilingual phrase table used to learning rules and that will pull on the performance down. In order to avoid a pessimistic result of low-coverage dictionary, on one hand, it is possible to deal with the unknown words like different lexical category, accordingly, as an example, a Kazakh-Turkish rule that apply on PRN UNKNOWN N patterns, it may carry out compromise between the pronoun and noun, though the word between them (perhaps a verb) is unknown, in contrast, it is possible to follow [126] approach, this concluded as follows: It is allowing for non-proficient professional users to insert entries into monolingual dictionaries especially those used in RBMT is introduced, when sentence contain unknown words, the user is requested to insert the new word in the dictionary, and it is also requested to insert it in the TL monolingual dictionary, that if the user known its translation. This method will help to alleviate impact of unknown words.

6. As a feature plan to continue development on the weighted transfer module that we built it to apply only chunker transfer rules (patterns of words), and that will be by extending that module to be applied into other stages of structural transfer such as interchunk and postchunk transfer rules too (patterns of chunks). Both of them are analogous to the chunker, but with some dissimilarity.

REFERENCES

[1] S. Tripathi and J. K. Sarkhel. Approaches to machine translation. *Annals of Library and Information studies*, 57:388–393, 2010.

[2] M. R. Costa-Jussà and M. Farrús. Statistical machine translation enhancements through linguistic levels: A survey. *ACM Computing Surveys (CSUR)*, 46:42, 2014.

[3] G. Thurmair. Comparing different architectures of hybrid machine translation systems. In *Proceedings of MT Summit XII*, pages 340–347, 2009.

[4] F. J. Och. Statistical machine translation: Foundations and recent advances. *Tutorial at MT Summit*, 2005.

[5] M. R. Costa-Jussa and J. A. Fonollosa. Latest trends in hybrid machine translation and its applications. *Computer Speech & Language*, 32(1):3–10, 2015.

[6] T. Ehara. System combination of rbmt plus spe and preordering plus smt. In *Proceedings of the 2nd Workshop on Asian Translation (WAT2015)*, pages 29–34, 2015.

[7] J. Hutchins, William and H. L. Somers. *An introduction to machine translation*, volume 362. Academic Press London, 1992.

[8] W. Weaver. Translation. *Machine translation of languages*, 14:15–23, 1955.

[9] M. Nagao, J.-i. Tsujii, K. Yada, and T. Kakimoto. An english japanese machine translation system of the titles of scientific and engineering papers. In *Proceedings of the 9th conference on Computational linguistics-Volume 1*, pages 245–252, 1982.

[10] J. Senellart, P. Dienes, and T. Varadi. New generation systran translation system. In *In Proceedings of MT Summit IIX Senellart J., Yang J., Rebollo A. 2003. SYSTRAN Intuitive Coding Technology. In Proceedings of MT Summit IX*, 2001.

[11] B. Scott and A. Barreiro. Openlogos mt and the sal representation language. 2009.

[12] A. Barreiro, B. Scott, W. Kasper, and B. Kiefer. Openlogos rule-based machine translation: Philosophy, model, resources and customization. *Machine Translation (MT)*, 25(*2*):107–126, 2011.

[13] A. Ranta. *Grammatical framework: Programming with multilingual grammars*, volume 173. CSLI Publications, Center for the Study of Language and Information Stanford, 2011.

[14] M. Silberztein. *La formalisation des langues: l'approche de NooJ*. ISTE Editions, 2015.

[15] S. Kim. Comma analysis and processing for improving translation quality of long sentences in rule-based english-korean machine translation. In *Proceedings of the 11th International Conference on Agents and Artificial Intelligence - Volume 2: ICAART*, pages 474–79, 2019.

[16] A. P. Mukta, A.-A. Mamun, C. Basak, S. Nahar, and M. F. H. Arif. A phrase-based machine translation from english to bangla using rule-based approach. In *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, pages 1–5, 2019.

[17] T. N. B. Son and P. Seresangtakul. Vietnamese-thai machine translation using rule-based. In*2017 9th International Conference on Knowledge and Smart Technology (KST)*, pages 187–92, 2017.

[18] A. Hurskainen and J. Tiedemann. Rule-based machine translation from english to finnish. In *Proceedings of the Second Conference on Machine Translation*, pages 323–29, 2017.

[19] S. Bayatli, S. Kurnaz, I. Salimzianov, J. N. Washington, and F. M. Tyers. Rule-based machine translation from kazakh to turkish. In *European Association for Machine Translation (EAMT)*, pages 49–58, 2018.

[20] I. Salimzyanov, J. N. Washington, and F. M. Tyers. A free/open-source Kazakh-Tatar machine translation system. In *Machine Translation Summit XIV*, pages 175–82, 2013.

[21] F. M. Tyers, J. N. Washington, I. Salimzyanov, and R. Batalov. A prototype machine translation system for Tatar and Bashkir based on free/open-source components. In *First Workshop on Language Resources and Technologies for Turkic Languages*, page 11, 2012.

[22] A. Sundetova, M. L. Forcada, and F. M. Tyers. A free/open-source machine translation system for English to Kazakh. In *3rd International Conference on Turkic Languages Processing (Turk-Lang 2015), Kazan, Tatarstan*, pages 78–91, 2015.

[23] K. Altıntaş. *Turkish to Crimean Tatar machine translation system*. PhD thesis, Bilkent. University, 2001.

[24] I. Hamzaoglu. Machine translation from Turkish to other Turkic languages and an implementation for the Azeri language, 1993.

[25] R. Gilmullin. The Tatar-Turkish machine translation based on the two-level morphological analyzer. In *Interactive Systems and Technologies: The Problems of Human-Computer Interaction*, pages 179–86, 2008.

[26] A. C. Tantuğ, E. Adalı, and K. Oflazer. A mt system from turkmen to turkish employing finite state and statistical methods. In *European Association for Machine Translation (EAMT)*, pages 459–65, 2007.

[27] A. C. Tantuğ and E. Adalı. *Machine translation between turkic languages*, pages 237–254. Springer, Cham, 2018.

[28] M. Orhun. *Machine translation from Uyghur to Turkish*. PhD thesis, Istanbul Technical University, 2010.

[29] U. Tukeyev, A. Melby, and M. Zhumanov Zh. Models and algorithms of translation of the Kazakh language sentences into English language with use of link grammar and the statistical approach. In *Proc. of IV Congress of the Turkic World Math. Society*, pages 1–3, 2011.

[30] G. Altenbek and W. Xiao-long. Kazakh segmentation system of inflectional affixes. In *Proceedings of CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pages 183–90, 2010.

[31] J. Washington, I. Salimzyanov, and F. M. Tyers. Finite-state morphological transducers for three kypchak languages. In *LREC*, 2014.

[32] J. Centelles and M. R. Costa-Jussa. Chinese-to-spanish rule-based machine translation system. In *Proceedings of the 3rd Workshop on Hybrid Approaches to Machine Translation (HyTra)*, pages 82–86, 2014.

[33] A. Toral, M. Ginestí-Rosell, and F. M. Tyers. An italian to catalan rbmt system reusing data from existing language pairs. In *Proceedings of the Second International Workshop on Free/Open-Source Rule-Based Machine Translation*, pages 77–81, 2011.

[34] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, pages 223–31, 2006.

[35] J. P. Turian, L. Shea, and I. D. Melamed. Evaluation of machine translation and its evaluation. Technical report, NEW YORK UNIV NY, 2006.

[36] A. L. Berger, P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. R. Gillett, J. D. Lafferty, R. L. Mercer, H. Printz, and L. Ureš. The candide system for machine translation. In *Proceedings of the workshop on Human Language Technology*, pages 157–62, 1994.

[37] D. Wu and W. Hkust, Hongsing. Machine translation with a stochastic grammatical channel. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 1408–15, 1998.

[38] E. Yıldırım and A. C. Tantuğ. The feasibility analysis of re-ranking for n-best lists on englishturkish machine translation. In *2013 IEEE INISTA*, pages 1–5, 2013.

[39] A. Eisele, C. Federmann, H. Saint-Amand, M. Jellinghaus, T. Herrmann, and Y. Chen. Using moses to integrate multiple rule-based machine translation engines into a hybrid system. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 179–82, 2008.

[40] Y. Chen and A. Eisele. Hierarchical hybrid translation between english and german. In *Proceedings of the 14th Annual Conference of the European Association for Machine Translation*, pages 90–97, 2010.

[41] N. Habash, D. Bonnie, and M. Christof. Symbolic-to-statistical hybridization: extending generation-heavy machine translation. *Machine Translation*, 23(1):23–63, 2009.

[42] E.-J. Park, O.-W. Kwon, K. Kim, and Y.-K. Kim. Classification-based approach for hybridizing statistical and rule-based machine translation. *ETRI Journal*, 37(3):541–50, 2015.

[43] A. Ahsan, P. Kolachina, S. Kolachina, D. Sharma, and R. Sangal. Coupling statistical machine translation with rule-based transfer and generation. *AMTA 2010 - 9th Conference of the Association for Machine Translation in the Americas*, 2010.

[44] D. Banik, S. Sen, A. Ekbal, and P. Bhattacharyya. Can smt and rbmt improve each other's performance?-an experiment with english-hindi translation. In *Proceedings of the 13th International Conference on Natural Language Processing*, pages 10–19, 2016.

[45] S. Appleby and P. M. Prol. The multilingual world wide web. *BT technology journal*, 18:71–72, 2000.

[46] P. Bhattacharyya et al. Comparison of smt and rbmt; the requirement of hybridization for marathi-hindi mt. *arXiv preprint arXiv:1703.03666*, 2017.

[47] M. Okpor. Machine translation approaches: issues and challenges. *International Journal of Computer Science Issues (IJCSI)*, 11:159, 2014.

[48] S. Dave, J. Parikh, and P. Bhattacharyya. Interlingua-based english–hindi machine translation and language divergence. *Machine Translation*, 16(4):251–304, 2001.

[49] B. Vauquois and C. Boitet. Automated translation at grenoble university. *Computational Linguistics*, 11:28–36, 1985.

[50] H.-D. Maas. The mt system susy. *Machine translation today: the state of the art, Edinburgh Information Technology Series*, 2:209–246, 1987.

[51] N. Ashraf and a. AhmadM. Experimental framework using web-based tools for evaluation of machine translation techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*, 6:223–228, 2016.

[52] P. Dirix, I. Schuurman, and V. Vandeghinste. Metis-ii: Example-based machine translation using monolingual corpora-system description. In *Proceedings of the Example-Based Machine Translation Workshop held in conjunction with the 10th Machine Translation Summit*, pages 43–50, 2005.

[53] P. Koehn. *Statistical machine translation*, volume 4. Cambridge University Press, 2009.

[54] F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational linguistics*, 29(1):19–51, 2003.

[55] F. J. Och and H. Ney. Statistical multi-source translation. In *Proceedings of MT Summit*, volume 8, pages 253–58, 2001.

[56] P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–11, 1993.

[57] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst. Moses: Open source toolkit for statistical machine translation. 2007.

[58] Y. Zhang. *Chinese-English Statistical Machine Translation by Parsing*. PhD thesis, University of Oxford, 2006.

[59] J. Tiedemann, F. Cap, J. Kanerva, F. Ginter, S. Stymne, R. Östling, and M. Weller-Di Marco. Phrase-based smt for finnish with more data, better models and alternative alignment and translation tools. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, volume 2, pages 391–98, 2016.

[60] J. Hutchins. Example-based machine translation: a review and commentary. *Machine Translation*, 19:197–211, 2005.

[61] N. Kalchbrenner and P. Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–09, 2013.

[62] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[63] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–34, 2014.

[64] C. Tillmann, S. Vogel, H. Ney, A. Zubiaga, and H. Sawaf. Accelerated dp based search for statistical translation. In *Fifth European Conference on Speech Communication and Technology*,1997.

[65] S. Bangalore, O. Rambow, and S. Whittaker. Evaluation metrics for generation. In *Proceedings of the first international conference on Natural language generation-Volume 14*, pages 1–8. Association for Computational Linguistics, 2000.

[66] S. Nießen, F. J. Och, G. Leusch, H. Ney, et al. An evaluation tool for machine translation: Fast evaluation for mt research. In *LREC*, 2000.

[67] W. Zhu, K. Papineni, S. Roukos, and T. Ward. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 311–318, 2001.

[68] G. Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc., 2002.

[69] M. R. Costa-Jussà, M. Farrús, J. B. Mariño, and J. A. Fonollosa. Study and comparison of rulebased and statistical catalan-spanish machine translation systems. *Computing and informatics*, 31(2):245–270, 2012.

[70] G. Klein, Y. Kim, Y. Deng, J. Senellart, and A. M. Rush. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*, 2017.

[71] P. Koehn, H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the association for computational linguistics companion volume proceedings of the demo and poster sessions*, pages 177–80, 2007.

[72] J. Tiedemann. Parallel data, tools and interfaces in opus. In *Lrec*, volume 2012, pages 2214–2218, 2012.

[73] M. Melero, A. Oliver, T. Badia, and T. Suñol. Dealing with bilingual divergences in mt using target language n-gram models. In *Proceedings of the METIS-II Workshop: New Approaches to Machine Translation*, pages 19–26, 2007.

[74] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710,1966.

[75] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

[76] M. Popović and H. Ney. Word error rates: Decomposition over pos classes and applications for error analysis. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 48–55. Association for Computational Linguistics, 2007.

[77] D. Coughlin. Correlating automated and human assessments of machine translation quality. In *Proceedings of MT summit IX*, pages 63–70, 2003.

[78] F. M. Tyers. *Feasible lexical selection for rule-based machine translation*. PhD thesis, 2013.

[79] C.-Y. Lin and F. J. Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting*

on Association for Computational Linguistics, page 605. Association for Computational Linguistics, 2004.

[80] C. Callison-Burch, M. Osborne, and P. Koehn. Re-evaluation the role of bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2006.

[81] M. Denkowski and A. Lavie. Challenges in predicting machine translation utility for human post-editors. In *Proceedings of AMTA 2012*, 2012.

[82] L. M. Paul, G. F. Simons, C. D. Fennig, et al. Ethnologue: Languages of the world. *Dallas, TX: SIL International. Available online at www. ethnologue. com/. Retrieved June*, 19:2011, 2009

[83] A. Göksel and C. Kerslake. *Turkish: A comprehensive grammar*. Routledge, 2004.

[84] J. Washington, I. Salimzyanov, and F. M. Tyers. Finite-state morphological transducers for three kypchak languages. In *LREC*, pages 3378–3385, 2014.

[85] R. Mukhamedova. *Kazakh: A comprehensive grammar*. Routledge, 2015.

[86] K. Lindén, E. Axelson, S. Hardwick, T. A. Pirinen, and M. Silfverberg. Hfst—framework for compiling and applying morphologies. In *International Workshop on Systems and Frameworks for Computational Morphology*, pages 67–85. Springer, 2011.

[87] C. Cöltekin. A freely available morphological analyzer for Turkish. In *LREC*, pages 19–28, 2010.

[88] K. Altintas and I. Cicekli. A morphological analyser for Crimean Tatar. In *Proceedings of the 10th Turkish Symposium on Artificial Intelligence and Neural Networks (TAINN'2001)*, pages 180–189, 2001.

[89] J. Washington, M. Ipasov, and F. M. Tyers. A finite-state morphological transducer for Kyrgyz. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*, pages 934–940, 2012.

[90] A. C. Tantuğ, E. Adalı, and K. Oflazer. Computer analysis of the Turkmen language morphology. In *Advances in Natural Language Processing*, pages 186–193. 2006.

[91] Ç. Çöltekin. A set of open source tools for Turkish natural language processing. In *LREC*, pages 1079–1086, 2014.

[92] F. M. Tyers and J. Washington. Towards a free/open-source universal dependency treebank for Kazakh. In *Proceedings of the 3rd International Conference on Computer Processing in Turkic Languages (TurkLang)*, pages 276–289, 2015.

[93] F. Karlsson, A. Voutilainen, J. Heikkilae, and A. Anttila. *Constraint Grammar: a languageindependent system for parsing unrestricted text*. Walter de Gruyter, 1995.

[94] F. M. Tyers, F. Sánchez-Martínez, and M. L. Forcada. Flexible finite-state lexical selection for rule-based machine translation. In *European Association for Machine Translation*, pages 213–220, 2012.

[95] M. L. Forcada, B. I. Bonev, S. O. Rojas, J. P. Ortiz, G. R. Sánchez, F. S. Martínez, C. Armentano-Oller, M. A. Montava, and F. M. Tyers. Documentation of the open-source shallow-transfer machine translation platform apertium. *Online] Departament de lenguatges i Sistemes Informatics Universitat d¨ Alacant, Available: http://xixona. dlsi. ua. es/˜fran/apertium2-documentation. pdf,[Accessed 27th April 2014]*, 2007.

[96] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

[97] C. D. Manning, C. D. Manning, and H. Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[98] F. M. Tyers, F. Sánchez-Martínez, and M. L. Forcada. Unsupervised training of maximumentropy models for lexical selection in rule-based machine translation. In *Proceedings of the 18th Annual Conference of the European Association for Machine Translation*, 2015.

[99] P. Koehn. Pharaoh: a beam search decoder for phrase-based statistical machine translation models. In *Conference of the Association for Machine Translation in the Americas*, pages 115–124, 2004.

[100] C. Dyer, J. Weese, H. Setiawan, A. Lopez, F. Ture, V. Eidelman, J. Ganitkevitch, P. Blunsom, and P. Resnik. cdec: A decoder, alignment, and learning framework for finite-state and context-free translation models. In *Proceedings of the ACL 2010 System Demonstrations*, pages 7–12, 2010.

[101] K. Heafield. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–97. Association for Computational Linguistics, 2011.

[102] A. Hurskainen and J. Tiedemann. Rule-based machine translation from english to finnish. In *Proceedings of the Second Conference on Machine Translation*, pages 323–29, 2017.

[103] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144– 152. ACM, 1992.

[104] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for computational Linguistics, 2003.

[105] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.

[106] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

[107] M. More and B. Tidke. A framework for summarization of online opinion using weighting scheme. *Adv Comp Intel*, 2(3):1–9, 2015.

[108] V. N. Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

[109] C. J. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[110] J. A. Suykens and J. Vandewalle. Least squares support vector machine classifiers. *Neural processing letters*, 9(3):293–300, 1999.

[111] J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin dags for multiclass classification. In *Advances in neural information processing systems*, pages 547–553, 2000.

[112] M. Murata, K. Uchimoto, Q. Ma, and H. Isahara. Using a support-vector machine for japaneseto- english translation of tense, aspect, and modality. *arXiv preprint cs/0112003*, 2001.

[113] S. Rajendran and B. Kalpana. Performance evaluation of kernels in multiclass support vector machines. *International Journal of Soft Computing and Engineering*, 1:138–145, 01 2011.

[114] T. Li, C. Zhang, and M. Ogihara. A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. *Bioinformatics*, 20(15):2429–2437, 2004.

[115] S. Knerr, L. Personnaz, and G. Dreyfus. Single-layer learning revisited: A stepwise procedure for building and training a neural network. In F. Fogelman Soulié and J. Hérault, editors, *Neurocomputing: Algorithms, Architectures and Applications*, volume F68 of *NATO ASI Series*, pages 41–50. Springer-Verlag, 1990.

[116] C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, 13(2):415–425, 2002.

[117] H. Daumé III. Support vector machines for natural language processing. *Lecture Notes*, 2004.

[118] B. Scholkopf. Support vector learning. *Ph. D. thesis, Technische Universitat Berlin.*, 1997.

[119] H.-T. Lin and C.-J. Lin. A study on sigmoid kernels for svm and the training of non-psd kernels by smo-type methods. *submitted to Neural Computation*, 3:1–32, 2003.

[120] C. Chang and C. Lin. Libsvm: A library for support vector machines, national taiwan university. *Taipei, Taiwan*, 2001.

[121] H.-T. Lin and L. Li. Support vector machinery for infinite ensemble learning. *Journal of Machine Learning Research*, 9(Feb):285–312, 2008.

[122] B. Zadrozny, J. Langford, and N. Abe. Cost-sensitive learning by cost-proportionate example weighting. In *ICDM*, volume 3, page 435, 2003.

[123] J. Milgram, M. Cheriet, and R. Sabourin. “one against one” or “one against all”: Which one is better for handwriting recognition with svms? 2006.

[124] M. L. Forcada, M. Ginestí-Rosell, J. Nordfalk, J. O’Regan, S. Ortiz-Rojas, P.-O. Juan, Antonio, F. Sánchez-Martínez, G. Ramírez-Sánchez, and F. M. Tyers. Apertium: a free/open-source platform for rule-based machine translation. *Machine Translation*, 25(2):127–44, 2011.

[125] Bayatli, S. (Sewale Bayatli), Kurnaz, S., Ali, A., Washington, J. N. and Tyers, F. M. (2020) "Unsupervised weighting of transfer rules in rule-based machine translation using a Maximum-Entropy approach". *Journal of Information Science and Engineering* 36 (1).

[126] V. M. Sánchez-Cartagena. Building machine translation systems for language pairs with scarce resources. 2015.

[127] F. Sánchez-Martínez and M. L. Forcada. Inferring shallow-transfer machine translation rules from small parallel corpora. *Journal of Artificial Intelligence Research*, 34:605–635, 2009.

APPENDIX A

APERTIUM: FREE/OPEN-SOURCE SHALLOW-TRANSFER MT

A.1 ARCHITECTURE OF THE APERTIUM MT

Apertium [124] is a free/open-source platform for building a classical shallow-transfer RBMT system consisting of a 10-modules (see Figure. A.1) Unix-style pipeline or assembly line, it has developed in 2005 via the Universitat d'Alacant. Modules can communicate between themselves using text streams that is for facilitating analysis and independent testing. This admits for using some of these modules separately from the rest of the MT system, for two functions such as natural-language processing tasks and research purposes.

A typical platform is speedy, it can translate thousands of words every second on a normal desktop computer; flexible in developing; and substantive, that make it achieve high performance without demand for existing data or huge parallel corpora for constructing it.

Apertium was basically designed for the Romance languages of the Iberian Peninsula, eventually has also been fitted for various, more remotely related, language pairs. This platform has next parts: machine translation engine, developer's tools, and linguistic data for an increasing number of language pairs and they are licensed beneath the free Software Foundation's General Public License (GPL).

A.2 MODULES IN THE APERTIUM PIPELINE

A.2.1 De-formatter

Separating the text for being able to be translated from the formatting tags. Formatting tags are deal with as "superblanks" because it is encapsulated in brackets and they placed between words in such a way that the rest of the modules see them as regular blanks.

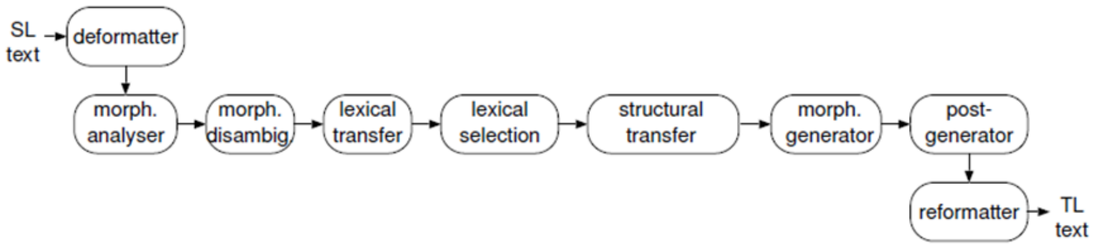


Figure A.1: The pipeline architecture of a typical Apertium MT system.

A.2.2 Morphological Analyzer

Segments the source-language (SL) text in surface forms (SF) (words, or, where detected, multiword lexical units) and for each, delivers one or more lexical forms (LF) consisting of lemma (dictionary or citation form), lexical category (or part-of-speech) and inflection information.

```
^досты/дос<n><acc>/дос<n>+лы<post>$
^құшақтау/құшақта<v><tv><ger><nom>$
^жақсы/жақсы<adj>/жақсы<adv>/жақсы<adj><advl>
     /жақсы<adj><subst><nom>/жақсы<adj>+е<cop><aor><p3><pl>
     /жақсы<adj>+е<cop><aor><p3><sg>
     /жақсы<adj><subst><nom>+е<cop><aor><p3><pl>
     /жақсы<adj><subst><nom>+е<cop><aor><p3><sg>$
^./.<sent>$
```

A.2.3 Morphological Disambiguator

A morphological disambiguator, in case of the Kazakh-Turkish translator based on the Constraint Grammar (CG) formalism [93], chooses the most acceptable sequence of morphological analyses for an ambiguous sentence.

```
^дос<n><acc>$
^құшақта<v><tv><ger><nom>$
^жақсы<adj>+e<cop><aor><p3><sg>$
^.<sent>$
```

A.2.4 Lexical Transfer

This module reads each SL LF and delivers the corresponding target-language (TL) LF by looking it up in a bilingual dictionary encoded as an finite-state transducer compiled from the corresponding XML file. The lexical transfer module may return more than one TL LF for a single SL LF.

```
^дос<n><acc>/dost<n><acc>/arkadaş<n><acc>$
^құшақта<v><tv><ger><nom>/kucakla<v><tv><ger><nom>$ ^жақсы<adj>/güzel<adj>/iyi<adj>$
^e<cop><aor><p3><sg>/i<cop><aor><p3><sg>$
^.<sent>/.<sent>$
```

A.2.5 Lexical Selection

A lexical selection module [94] chooses, based on context rules, the most adequate translation of ambiguous SL LFs.

```
^дос<n><acc>/dost<n><acc>/arkadaş<n><acc>$
^құшақта<v><tv><ger><nom>/kucakla<v><tv><ger><nom>$ ^жақсы<adj>/güzel<adj>/iyi<adj>$
^e<cop><aor><p3><sg>/i<cop><aor><p3><sg>$
^.<sent>/.<sent>$
```

A.2.6 Structural Transfer

The structural transfer module (as described in this thesis, see Chapter 3) applies a sequence of one or more finite-state constraint rules on the output of the lexical selection module so that it can select the left-right longest matching translation.

```
^default<default>{^dost<n><acc>$}$
^v<SV>{^kucakla<v><tv><><nom>$}$
^adj<SADJ>{^güzel<adj>+i<cop><aor><p3><sg>$}$
^sent<SENT>{^.<sent>$}$
```

A.2.7 Morphological Generator

It transforms the sequence of target–language LFs, produced by the structural transfer, to a corresponding sequence of target–language SFs.

dostu kucaklamak güzel.

A.2.8 Post-generator

Performs orthographic operations, for example elision (such as da + il = dal in Italian). This module has not been employed in our translator so far.

A.2.9 Reformatter

The reformatter works opposite the deformatter which is de-encapsulates any format information. The Apertium platform provides what can be called a ‘vanilla’ program and a formalism for describing linguistic data (if the module in question requires it) for each of the modules. We want to emphasis though that modules of the pipeline just described are independent from each other and thus can rely on different programs, different formalisms, and be of rule-based statistical or hybrid nature. For example, Constraint Grammar-based morphological disambiguator can be considered a drop-in replacement for the Hidden Markov Model-based statistical tagger found in a few other Apertium MT systems. So are the formalisms used for morphological transducers which are described next.

APPENDIX B

SOFTWARE RELEASED AS PART OF THIS THESIS

B.1 APERTIUM-TRANSFER-TOOLS

The software has been explained in this dissertation is executed in the apertium-transfer-tool package, which is released under the GNU GPL licence version 2.0 or later; it is possible be downloaded from https://github.com/sevilaybayatli/apertium-ambiguous. It is meant to replace a previous version of the apertium-transfer-tools package initially released by [127]. The apertium-transfer-tools generate a shallow-transfer rules encoded in the Apertium XML format (described in Appendix A).

A compatible version of the YASMET maximum-entropy toolkit is also provided as assistance for the user.