A Machine Assisted Human Translation System for Technical Documents

Vishwajeet Kumar vishwajeetkumar86@gmail.com
Ashish Kulkarni kulashish@gmail.com
Pankaj Singh pr.pankajsingh@gmail.com
Ganesh Ramakrishnan ganesh@cse.iitb.ac.in
Department of Computer Science and Engineering, IIT Bombay, Mumbai, India

Ganesh Arnaal ganesh.arnaal@smcapitaladvisors.in
SM Capital Advisors,Mumbai,India

Abstract

Translation systems are known to benefit from the availability of a bilingual lexicon for a domain of interest. A system, aiming to build such a lexicon from source language corpus, often requires human assistance and is confronted by conflicting requirements of minimizing human translation effort while improving the translation quality. We present an approach that exploits redundancy in the source corpus and extracts recurring *patterns* which are : *frequent, syntactically well-formed, and provide maximum corpus coverage*. The patterns generalize over phrases and word types and our approach finds a succinct set of good patterns with high coverage. Our interactive system leverages these patterns in multiple iterations of translation and post-editing, thereby progressively generating a high quality bilingual lexicon.

1 Introduction

The problem of language translation has been in focus for many decades and has seen contributions from both linguistic and computer science communities. Linguistic contribution (Streiter (1996)) has come in the form of several language resources comprising of dictionaries, grammar and studies on units of translation. Computer science community has contributed in coming up with formal machine translation (MT) models (Vogel et al. (2003)) that leverage corpus statistics along with linguistic features and resources. There is a body of work (Federico et al. (2014); Alabau et al. (2014)) that studies the complementary contributions of humans and MT models and present "machine-centric" translation systems that leverage human input. These systems, referred to as computer aided translation (CAT) systems, typically employ a statistical MT model to translate text and provide a post-editing tooling to enable humans to correct the resulting translations. Human feedback and corrections are used to adapt and retrain the translation model. What constitutes the right unit of translation and how can the human feedback be incorporated in the underlying translation model, pose interesting research challenges.

A domain corpus is often replete with redundancy arising due to the choice of vocabulary and syntax. Translation memory-based systems (Sato and Nagao (1990)) exploit this redundancy and store recurring phrases and their translations. We are fur-

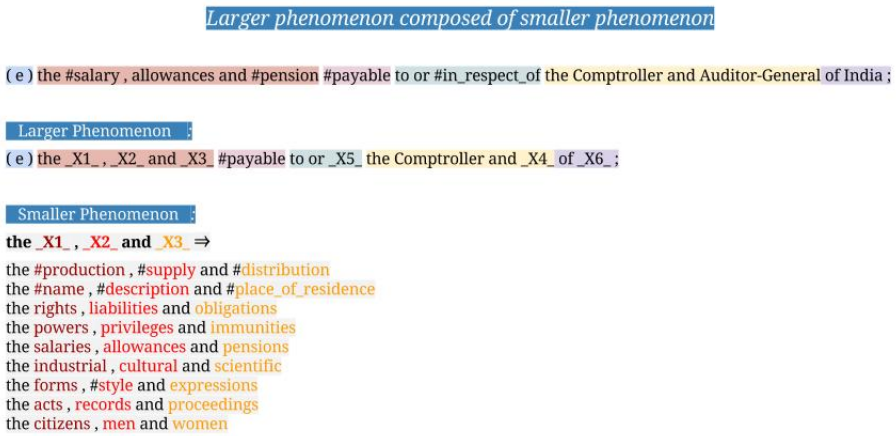


Figure 1: An example illustrating the principle of compositionality and higher order patterns in a domain corpus

ther motivated by Frege's principle of compositionality (Pelletier (1994)), which states that the meaning of a compound expression is a function of the meaning of its parts and of the syntactic rule by which they are combined. Figure 1 shows an example, taken from legal domain, of a compound expression and its constituent expressions. Some of these expressions comprise of categories that generalize over several tokens, thus, forming higher order recurring patterns in the corpus. Extraction of these patterns and using them as the unit of translation might enable us to better capture the structure and semantics of the domain.

An in-domain (especially technical, legal) corpus often adheres to a certain lexical and syntactic structure and is often less amenable to creative or "free" translation. These domains, therefore, might be good candidates for translation using rule-based systems Terumasa (2007), comprising of source and target language dictionaries, grammars and translation rules. Grammatical Framework (GF) (Ranta (2004)) provides the necessary formalism to theorize rule-based translations and also provides a system to author abstract and concrete language syntax.

We present an approach and a system that builds on these ideas to extract meaningful patterns from a domain corpus, gather human feedback on their translation and learn a rule-based translation system using the GF formalism. The system is "human-centric", in that, it heavily relies on manually curated linguistic resources, while the machine continuously prompts the human on *what* to translate. This interactive human-machine dialog produces a translation system that aims to achieve high precision in-domain translations and might find application in several technical domains including medical, education, legal etc.The system is available for demo at http://mtdemo.hostzi.com.

2 Related Work

There has been a lot of research on automated statistical machine translation (SMT) and several systems (Wang and Waibel (1998); Vogel et al. (2003); Och and Ney (2000);

Koehn et al. (2007)) have been proposed. While they are all typically based on a combination of a translation model and the target language model, the difference lies in their units of translation (word-based, phrase-based *etc.*) and translation decoding. The statistical approach to MT itself falls under the general category of example-based MT (EBMT) (Somers (1999)) or memory-based MT (Sato and Nagao (1990)). These approaches rely on the availability of a corpus or a database of already translated examples, and involve a process of matching a new input against this database to extract suitable examples and then determine the correct translation. These corpus-based approaches suffer from two major drawbacks - (1) parallel corpus is often expensive to generate and is often scarce or unavailable for certain language pairs or domains; (2) their quality of translation is not as good as that of human translation and therefore not suitable for certain applications like those involving translation of government documents or academic books.

Rule-based machine translation systems (RBMT) like Apertium (Forcada et al. (2011)) alleviate the need for a sentence aligned parallel corpus but require explicit linguistic data in the form of morphological and bilingual dictionaries, grammars and structural transfer rules. Apertium is a free and open-source machine translation platform with liguistic data for a growing number of language pairs along with the necessary tools and a translation engine. However, these systems typically involve a complex pipeline and statistical tools, making it difficult to track and correct errors.

Many researchers in the past have claimed and suggested that we cannot remove humans completely from the translation pipeline (Kay (1980)). In order to cater to applications requiring a high-quality translation, the output of MT systems is often revised by a *post-editing* phase. Several computer-aided translation (CAT) tools exist that are either desktop-based (Carl (2012); Aziz et al. (2012)), iOmegaT[1] or web-based (Federico et al. (2014); Denkowski and Lavie (2012); Roturier et al. (2013)). As an alternative to pure post-editing systems, interactive machine translation (IMT) (Toselli et al. (2011)) combines a MT engine with human, in an interactive setup, where, the MT engine continuously exploits human feedback and attempts to improve future translations. Daniel Ortiz-Martınez (2011); Ortiz-Martínez et al. (2010), for instance, talk about online learning in the machine translation pipeline, where, human feedback on translations is used to re-estimate the parameters of a statistical machine translation model. Bertoldi et al. (2013) address the problem of dynamically adapting a phrase-based SMT model from user post-editing by means of a caching mechanism. Their cache-based model combines a large global static model with a small local and dynamic model estimated from recent items in the input stream. Lavie (2014) incorporate human feedback and propose three online methods for improving an underlying MT engine based on translation grammar, Bayesian language model and parameter optimization. Anusaarka (Bharati et al. (2003)), a hybrid machine translation system for English to Hindi, also involves interaction but is restricted to authoring rules for word sense disambiguation. Ranta had proposed Grammatical Framework (GF) (Ranta (2004)) which is a grammar formalism and a programming language for multilingual grammar applications. One good example of applications using GF[2] is Molto (Cristina Espa˜na-Bonet (2011)), a machine translation system for patent translation.

While our approach builds on existing work, our primary contribution is a framework and a system for high quality domain corpus translation. Our system gathers manual translation of redundant patterns in an interactive setting and uses these to

[1] http://www.omegat.org/

[2] http://www.grammaticalframework.org/

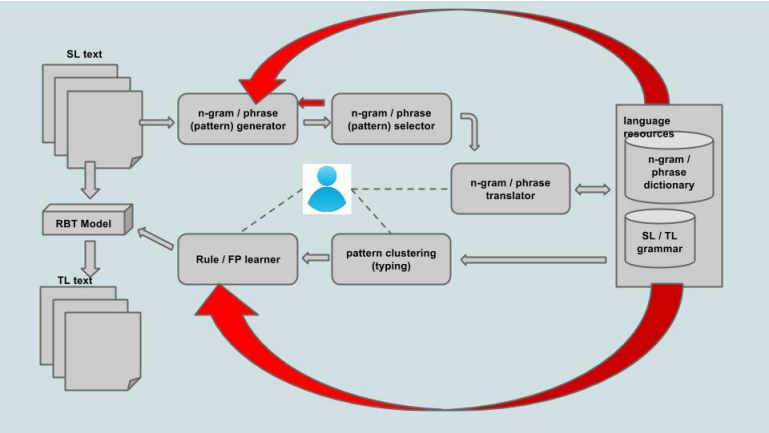


Figure 2: System Architecture

build language resources like grammars and bilingual lexicons. These are realized using the GF formalism and the translation system continues to benefit from more human feedback.

3 System Architecture

Our system follows an iterative pipeline architecture where every component is modular. The system is interactive and takes human feedback on translations. The feedback is used to build linguistic resources and is incorporated into the underlying translation model. The translation model itself is expressed using the grammatical framework formalism, which is based on functional programming and type theory. This expressivity and abstraction makes the model easily programmable by humans.

3.1 Pattern Extraction

This module captures redundant translation units present in the corpus. It takes as input a domain specific corpus and monolingual typed dictionaries and produces frequently occurring translation units as output. It uses frequent pattern mining technique to capture exhaustive set of frequent translation units. In order to extract more general translation units, we extract patterns with gaps where a gap might be of varying length. A gap could be considered as a generalized form of an entity and is represented as "X" or "_X_". The length of the gap controls the generalization. As output, the module produces a directed acyclic graph of frequent translation units in the corpus. Algorithm 1 contains details of our frequent pattern mining algorithm. The module also supports filtering of invalid translation units. An invalid translation unit is the one that does not honor pattern compositionality.

3.2 Pattern Selection

Pattern Extraction (Section 3.1) mines a large number of redundant patterns as potential translation units. Since getting manual translations for these candidate translation units is a costly operation, we identify a subset of patterns that are both diverse and maximally cover the in-domain source language corpus. The pattern selection algorithm (Refer Algorithm 2) provides details on this selection of a subset of "good" patterns, where, goodness of the subset is measured in terms of corpus

Algorithm 1: Algorithm: FPM algorithm

Data: Corpus C, Pattern length L, Frequency threshold T, Maximum consecutive gaps of tokens G
Result: Set F of frequent patterns
Maintain a dictionary structure globalPatternList where key is pattern and value is list of span
for *each sentence s in C* **do**
  maintain an array of list, slist, of size $|s|$, such that, $slist[i]$ stores all one length pattern along with its span in the sentence which starts from s_i
  using slist, create a 2D array of list, smatrix, of size $|s|xL$ such that, $smatrix[i][j]$ stores all patterns, along with its span in s, which starts from s_i and of pattern length j
  Filter pattern from smatrix whose span is syntactically incomplete
  Add these patterns to globalPatternList
end
Initialize patternWithGap dictionary
for *i in* $1 \cdots L$ **do**
  for *valid mask v of length L* **do**
    for *pattern p of length i in globalPatternList* **do**
      apply v on pattern p and create a new pattern $\hat{p}$
      if $\hat{p}$ *is present in patternWithGap* **then**
        update its spanlist by doing union with span list of p
      else
        add $\hat{p}$ in patternWithGap with its spanlist as spanlist of p
      end
    end
    remove patterns of length i and with gap position according to mask and having spans count less than T
  end
end
remove patterns from globalPatternList whose number of spans is below **T**
output $patternWithGap \cup globalPatternList$

coverage. Figure 3 provides an example, where, the first column contains sample text from a corpus and the other columns show the extracted patterns and the patterns (in bold) after the selection step.

3.3 Pattern Translator

Translator module involves users to provide translations of translation units. In this module five system generated translations are displayed to translator out of which he can select best translation for a particular translation unit or he can even write a new translation.

3.4 Generalization of Translation Units

At each iteration we identify important non-terminals present at that level and use this information while generating translation units at the next level. This module helps in generalizing translation units by clustering them together. This in turn helps in reducing

Sample input text	Sample patterns subset		
	on the expiration of the X period	the fixed period	one year
on the expiration of every second year in accordance	on the expiration of the said period	***the X period(5)***	**his term of office(4)**
on the expiration of every second year in accordance	on the expiration of the fixed period	a period of NP	his term
on the expiration of the said period	on the expiration of a period of ten years	a period of ten years	
on the expiration of the fixed period	on the expiration of every second year in accordance	a period	
on the expiration of his term of office	on the expiration of a period of six months	ten years	
on the expiration of his term of office	on the expiration of a period of one year	every second year	
on the expiration of a period of ten years	on the expiration of his term of office	***every second year in accordance(3)***	
on the expiration of a period of six months	***on the expiration of a period of NP (1)***	a period of six months	
on the expiration of a period of six months	***on the expiration(2)***	six months	
on the expiration of a period of six months	the said period	a period of one year	
on the expiration of a period of one year			

Figure 3: Example

the number of rules required to express compositionality. In terms of grammar, we can think of it as identifying LHS of productions. Arguably, this module must also serve the purpose of organizing non-terminals such that it is useful for translation task. Since we are identifying domain specific concepts (non-terminals) which can be translated, it must also keep the target language in mind.
We have observed in various sentences that if internal reordering[3] of phrases in sentences having same cannonical structure is same then their external reordering[4] also remains same. So we tried to cluster phrases having same internal reordering into one cluster. It is very clear from the objective of this module that clustering of translation units should be based on some translation in-variance phenomenon. Since the group represent all the translation units present in that group, it should also represent their translation behavior. Same external reordering help a category to generalize these translation units for higher level translation unit generation while same internal re-ordering will help in writing single translation rule for all the member translation units. We used reordering distortion score between translations of two translation units as a measure to cluster translation units.

3.5 Rule/FP Learner

Once patterns are extracted, selected, translated and stored in database, we annotate sentences with pattern name or in other words represent sentences in the form of sequence of translation units. If a sentence is completely covered by the set of patterns, it can be represented in terms of patterns. Once a sentence is represented in such a canonical form, we parse and linearize it using grammatical framework rules.
Idea of using functional programming and type theory in machine translation came from logical framework and ALF[5]. The logical framework ALF was based on the constructive type theory of Martin-Löf (Martin-Löf 1984, Nordström & al. 1990). Constructive type theory has also proven usable for meaning representation in natural languages (Ranta 1994). Logical frameworks were used to define logic in other perspectives but logic in machine translation means grammar. The type checking and proof search machinery provided by a logical framework like ALF gives tools for the kind of semantic analysis needed in machine translation. And here the missing component was parsing and linearization which was provided by Grammatical framework developed by Arne Ranta.

Grammatical framework is nothing but an extension of logical framework with a component called concrete syntax. Reordering rules and rules for handling gender, number and person information while doing look up is written using grammatical framework. The main purpose behind using grammatical framework is its functional nature. Gram-

[3]Reordering of tokens within a pattern during its translation from source to target language
[4]Reordering of a pattern within a sentence during the translation of that sentence
[5]ALF (Another Logical Framework) is a logical framework based on Martin-Lof type theory

Algorithm 2: Pattern Selection

Data: Dictionary of patterns P with its spanslist, Number of words in corpus N, Max size of selected set k
Result: Set F of diverse and high coverage (in terms of words) patterns

$F = \emptyset$
$bitCorpus \leftarrow \emptyset$
for $i \leftarrow 1$ ***to*** N **do**
  $bitCorpus[i] \leftarrow false$
end
for $i \leftarrow 1$ ***to*** k **do**
  $currentBest \leftarrow NULL$
  $currentBestCoverage \leftarrow 0$
  for *each pattern* p *in* $P \setminus F$ **do**
    $coverage_p \leftarrow 0$
    $coverage_p \leftarrow$ count of false bits in bitCorpus which is in spanlist of p
    if $coverage_p > currentBestCoverage$ **then**
      $currentBest = p$
      $currentBestCoverage = coverage_p$
    end
  end
  if $currentBest$ **then**
    $F \leftarrow F \cup$currentBest)
    set $BitCorpus[i]$ = true if i is in the spanlist of currentBest
  else
    break
  end
end
output F

matical framework also has a concept called abstract syntax which provides interlingua representation. Interlingua representation helps in linearizing in different languages very easily just by writing concrete grammar for that language.

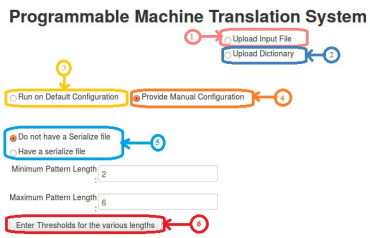


Figure 4: Interactive user interface for providing parameters to Frequent Pattern Miner

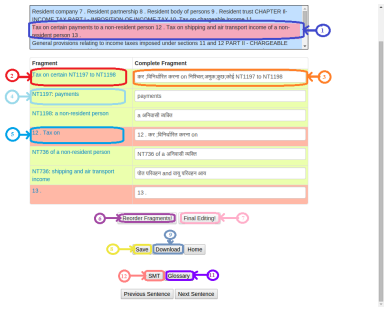


Figure 5: Interactive user interface for humans to translate patterns and n-grams.

3.6 System User Interface

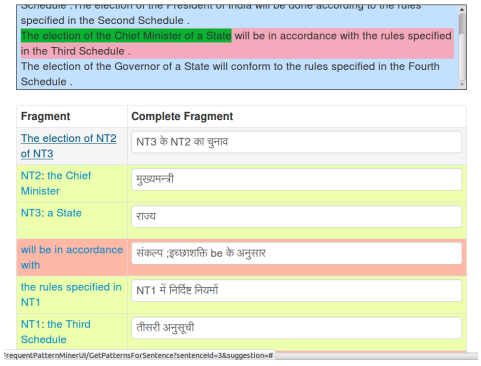
Our system has a highly interactive user interface for humans to translate patterns and n-grams. It also has provision for expert users to configure pattern length and frequency threshold for pattern extraction. Figure 4 depicts the features provided to expert users. Users can upload a new corpus using the *Upload Input File* option marked with label 1 in the figure. The *Upload Dictionary* option (labeled 2) enables users to upload bilingual dictionaries for the system to perform lookups and provide translation suggestions. Users can either choose to run the system and extract patterns on the optimized default configuration (labeled 3) or they can manually configure the pattern length and frequency (labeled 4).

Once patterns are extracted, filtered and validated by the system, users use the web-based system shown in the Figure 5 for providing translation feedback. Human translators are shown the current sentence (labeled 1) along with the previous and next sentences as context information. Patterns are displayed below column labeled *fragment* (label 2). On hover over patterns or untranslated n-grams, the span covering that pattern or n-gram in the sentence gets highlighted (refer to figure 6a). For patterns containing generalized non terminals (labeled 2), translators can view all the instances of non terminals by hovering over the NTs. Instance of a non terminal is represented by label 4 in Figure 5. Initially a translation of patterns and untranslated n-grams (labeled 5) is suggested by the system using translated patterns database, glossary look-up and SMT. Translators can even configure the source for getting the suggestion (a) they can choose to get translation suggestion from SMT system by clicking on SMT button (labeled 12) or (b) they can choose to get translation suggestions from database by clicking on glossary button (labeled 11). Translators can edit the translation suggestion (labeled 3) given by the system and correct them. They can also reorder the composed translation of sentence by clicking on reorder button (labeled 6), which presents a simple drag and drop interface to the user (refer to Figure 6b). Finally, if user wish to edit the composed translation they can do that by clicking on final editing button depicted by label 7 in Figure 5. After final editing, users can save the translation by clicking on save button (labeled 8). Users can also download the translations by clicking on the *download* button (labeled 9). In order to get translation suggestions for a particular word or phrase, users can enter the text in *suggestions* panel on the right and get multiple translation suggestions for the particular word or phrase.
Important Features of the system:-

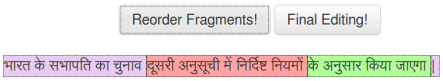
- Once a translator translates a pattern, a pattern instance or an n-gram, the system auto-translates it if next time it appears in a sentence.
- If a pattern, pattern instance or n-gram is translated differently in different sentences, the system lists all of them as choices for the user to choose from or enter a new translation.
- The system also has an integrated suggestion component that fetches translation suggestions from various sources. Users can use this to get translation suggestions for words or phrases and choose the best translation from the choices.

4 Evaluation

We evaluate the system in terms of the quality of extracted patterns, GF grammar and system efficiency. Evaluation was done on five public datasets *viz.* the Constitution of

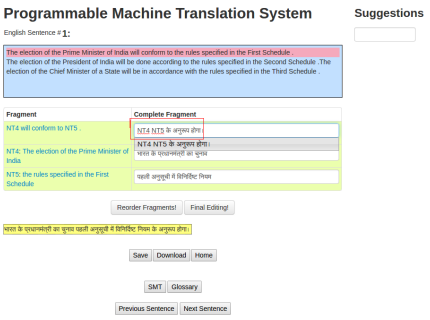


(a) On hover over patterns the part of sentence covering the pattern gets highlighted

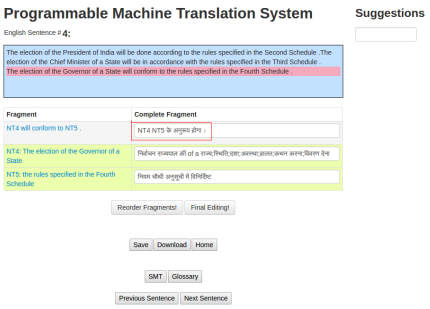


(b) Reordering composed translation of sentence

(c) Final editing



(d) Pattern translated by human highlighted in red rectangle



(e) Same pattern appearing again in another sentence

Figure 6: Illustration of various features of the system user interface

India[6], Spoken Tutorial[7], NCERT Biology[8], Income-tax Act[9], and NCERT Physics[10]. These datasets belong to the domains of government documents, technical tutorials and academic books, where, high quality translations are an imperative. Table 1 shows the corpus statistics in terms of number of sentences for each of the datasets.

4.1 Number of Frequent Patterns and Corpus Coverage

Number of Frequent patterns increase as the size of corpus increases. Corpus coverage depends on the number of syntactically well formed patterns extracted from the corpus which adhere to specified pattern length and frequency. Table 10 depicts information about number of filtered patterns extracted and coverage on five different corpus.

[6] http://indiacode.nic.in/coiweb/welcome.html
[7] http://spoken-tutorial.org/
[8] http://www.ncert.nic.in/NCERTS/textbook/textbook.htm?kebo1=0-22
[9] http://www.incometaxindia.gov.in/pages/acts/income-tax-act.aspx
[10] http://www.ncert.nic.in/NCERTS/textbook/textbook.htm?leph1=0-8

Table 1: Datasets and corpus coverage by patterns

Domain	#Sentences	#Frequent Patterns	#Frequent Instances	#Coverage %
Constitution of India	1582	12946	154218	86.62
Spoken Tutorial	16233	32974	10846	78.32
NCERT Biology	1144	615	12407	60.82
Income-Tax Act	1758	8391	104998	89.34
NCERT Physics	8013	15070	244034	79.94

4.2 Effect of Varying Pattern Length and Frequency Threshold for Pattern Extraction

One of the criterion to assess the quality of an individual extracted pattern is whether or not it appears in unseen data, thereby covering sentences in that data. A set of such patterns is then considered to be "good" if it collectively offers a high coverage on an unseen data. We split the datasets into MINE and TEST, where, the MINE split was used for extracting patterns and their coverage (in terms of number of words covered) was evaluated on the TEST split. We perform three-fold cross validation, varying both pattern length and frequency threshold from 2 to 6 and report coverage on MINE and TEST sets. Figure 7 captures the trade-off between pattern length, frequency threshold and coverage. For a fixed threshold, as the pattern length increases, the coverage on both MINE and TEST sets progressively decreases. Same observation applies when we fix the pattern length and increase the frequency threshold. We also observe that the gap in coverage is much smaller for varying frequency thresholds at smaller lengths and this gap progressively widens as the pattern length increases.

4.3 Effect of Varying Dictionary Size on Corpus Coverage

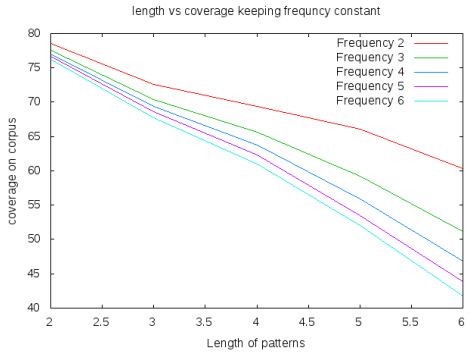
Our pattern selection algorithm constrains the cardinality of the set while maximizing a quality criteria like corpus coverage. Constraining the cardinality of the final set corresponds to limiting the size of the bilingual dictionary and this is desirable as the size of the bilingual dictionary is proportional to the human effort for translation. The corpus coverage increases with increasing size of the dictionary, however this increase is not linear but rather diminishes with increasing size of the dictionary. Figure 7d captures this relationship between coverage and fraction of patterns selected after sub-setting for different datasets.

4.4 Induced GF grammars

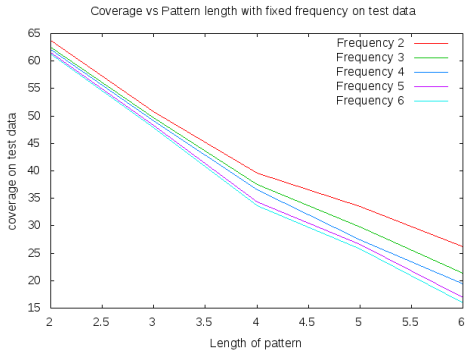
Once users provide translations of patterns, their instances, uncovered n-grams in sentences and reorders different chunks, grammatical framework rules are induced. Firstly, abstract syntax is induced which defines what meanings can be expressed in the grammar and then concrete English and concrete Hindi syntax is induced which provides mapping from meanings to strings in English and Hindi languages. Figure 8 illustrates a sample induced GF grammar. For a new sentence, extracted and translated patterns are given as input to GF grammars and if a match is found, then the sentence is reordered using the mapping from the concrete syntax. A more detailed example is available at `http://www.cse.iitb.ac.in/~vishwajeet/gf_rules.html`.

4.5 Conclusion

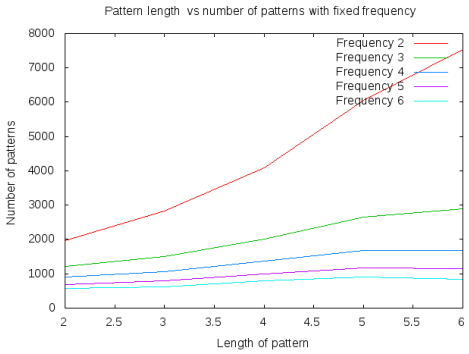
We presented an interactive machine translation approach for high quality translation of technical domain corpora. Given an in-domain source corpus, our system mines minimal



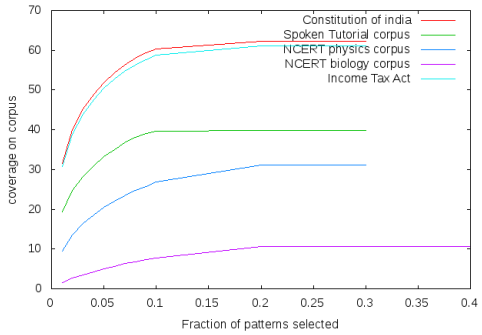
(a) Coverage vs. pattern length on the mining data



(b) Coverage vs. pattern length on the test data



(c) Pattern length vs number of patterns for a fixed threshold



(d) Coverage vs number of pattern selected after pattern selection

Figure 7: Corpus coverage for varying pattern lengths and frequency and coverage vs number of patterns selected after pattern selection

number of frequent patterns that maximally cover the corpus. Leveraging humans for their high quality translations, we continuously rebuild a rule-based translation engine that is realized using GF formalism.

```
abstract ST = {

flags coding = utf8 ;

flags startcat = Sent;

cat
    Sent; Sent1; NT2233; NT2234; P2; Ch1; Sent2; Ch2; Sent3; Sent4; Ch3; NT3402; Ch4; Ch5; P3; Sent5; P1;

fun
    NT2233_Fun : NT2233;
    NT2234_Fun : NT2234;
    Sent1_Fun : P1 -> Sent1;
    Sentence1_Fun : Sent1 -> Sent;
    Sent2_Fun : P2 -> Ch1 -> Sent2;
    Sentence2_Fun : Sent2 -> Sent;
    P2_Fun : P2;
    Ch1_Fun : Ch1;
    Sentence3_Fun : Sent3 -> Sent;
    Sent3_Fun : Ch2 -> Sent3;
    Ch2_Fun : Ch2;
    Sentence4_Fun : Sent4 -> Sent;
    Sent4_Fun : Ch3 -> Sent4;
    Ch3_Fun : Ch3;
    NT3402_Fun : NT3402;
    Sentence5_Fun : Sent5 -> Sent;
    Sent5_Fun : Ch4 -> P3 -> Ch5 -> Sent5;
    Ch4_Fun : Ch4;
    P3_Fun : NT3402 -> P3;
    Ch5_Fun : Ch5;
    P1_Fun : NT2233 -> NT2234 -> P1;

}
```

(a) Abstract Grammar

```
concrete STEng of ST = {

flags coding = utf8 ;

lincat
    NT2233 = Str;
    NT2234 = Str;
    Sent1 = Str;
    Sent = Str;
    P2 = Str;
    Ch1 = Str;
    Sent2 = Str;
    Ch2 = Str;
    Sent3 = Str;
    Sent4 = Str;
    Ch3 = Str;
    NT3402 = Str;
    Ch4 = Str;
    Ch5 = Str;
    P3 = Str;
    Sent5 = Str;
    P1 = Str;

lin
    NT2233_Fun  = "the spoken tutorial on Logical operators";
    NT2234_Fun  = "C and C++";
    Sent1_Fun p1 = p1;
    Sentence1_Fun sent1 = sent1;
    P2_Fun  = "In this tutorial";
    Ch1_Fun  = "we will learn about Logical operators like : Logical AND e.g. expression1 && expression2";
    Sent2_Fun p2 ch1 = p2 ++ ch1;
    Sentence2_Fun sent2 = sent2;
    Sentence3_Fun sent3 = sent3;
    Sent3_Fun ch2 = ch2;
    Ch2_Fun  = "Logical OR eg . expression1 OR expression2";
    Sentence4_Fun sent4 = sent4;
    Sent4_Fun ch3 = ch3;
    Ch3_Fun  = "Logical NOT eg . not ( Expression1 )";
    NT3402_Fun  = "the help of examples";
    Sentence5_Fun sent5 = sent5;
    Sent5_Fun ch4 p3 ch5 = ch4 ++ p3 ++ ch5;
    Ch4_Fun  = "We";
    P3_Fun nt3402 = "will do this with" ++ nt3402;
```

(b) Concrete English Grammar

```
concrete STHin of ST = {

flags coding = utf8 ;

lincat
    NT2233 = Str;
    NT2234 = Str;
    Sent = Str;
    Sent1 = Str;
    P2 = Str;
    Ch1 = Str;
    Sent2 = Str;
    Ch2 = Str;
    Sent3 = Str;
    Sent4 = Str;
    Ch3 = Str;
    NT3402 = Str;
    Ch4 = Str;
    Ch5 = Str;
    P3 = Str;
    Sent5 = Str;
    P1 = Str;

lin
    NT2233_Fun  = "तार्किक ऑपरेटर्स पर स्पोकन ट्यूटोरियल";
    NT2234_Fun  = "सी और सी++";
    Sent1_Fun p1 = p1;
    Sentence1_Fun sent1 = sent1;
    P2_Fun  = "इस ट्यूटोरियल में";
    Ch1_Fun  = "हम तार्किक ऑपरेटर्स के बारे में सीखना होगा जैसे : तार्किक AND उदाहरण: अभिव्यक्ति1 && अभिव्यक्ति2 ";
    Sent2_Fun p2 ch1 = p2 ++ ch1;
    Sentence2_Fun sent2 = sent2;
    Sentence3_Fun sent3 = sent3;
    Sent3_Fun ch2 = ch2;
    Ch2_Fun  = "तार्किक OR  जैसे: अभिव्यक्ति1 || अभिव्यक्ति2 ";
    Sentence4_Fun sent4 = sent4;
    Sent4_Fun ch3 = ch3;
    Ch3_Fun  = "तार्किक NOT उदाहरण: ! ( अभिव्यक्ति1 ) ";
    NT3402_Fun  = "उदाहरणो की सहायता";
    Sentence5_Fun sent5 = sent5;
    Sent5_Fun ch4 p3 ch5 = ch4 ++ p3 ++ ch5;
    Ch4_Fun  = "हम";
```

(c) Concrete Hindi Grammar

Figure 8: Induced abstract grammar, concrete english grammar and concrete hindi grammar

References

Alabau, V., Buck, C., Carl, M., Casacuberta, F., Garcıa-Martınez, M., Germann, U., González-Rubio, J., Hill, R., Koehn, P., Leiva, L., et al. (2014). Casmacat: A computer-assisted translation workbench. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 25–28.

Aziz, W., Castilho, S., and Specia, L. (2012). Pet: a tool for post-editing and assessing machine translation. In *LREC*, pages 3982–3987.

Bertoldi, N., Cettolo, M., and Federico, M. (2013). Cache-based online adaptation for machine translation enhanced computer assisted translation. *Proceedings of the XIV Machine Translation Summit*, pages 35–42.

Bharati, A., Chaitanya, V., Kulkarni, A. P., Sangal, R., and Rao, G. U. (2003). Anusaaraka: overcoming the language barrier in india. *arXiv preprint cs/0308018*.

Carl, M. (2012). Translog-ii: a program for recording user activity data for empirical reading and writing research. In *LREC*, pages 4108–4112.

Cristina Espa˜na-Bonet, Ramona Enache, A. S. A. R. L. M. M. G. (2011). Patent translation within the molto project. *MT Summit*.

Daniel Ortiz-Marfınez, Luis A. Leiva, V. A. s. G.-V. F. C. (2011). An interactive machine translation system with online learning. *ACL-HLT*, pages 68–73.

Denkowski, M. and Lavie, A. (2012). Transcenter: Web-based translation research suite. In *AMTA 2012 Workshop on Post-Editing Technology and Practice Demo Session*, page 2012.

Federico, M., Bertoldi, N., Cettolo, M., Negri, M., Turchi, M., Trombetti, M., Cattelan, A., Farina, A., Lupinetti, D., Martines, A., et al. (2014). The matecat tool. In *Proceedings of COLING*, pages 129–132.

Forcada, M. L., Ginestí-Rosell, M., Nordfalk, J., O'Regan, J., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Sánchez-Martínez, F., Ramírez-Sánchez, G., and Tyers, F. M. (2011). Apertium: a free/open-source platform for rule-based machine translation. *Machine translation*, 25(2):127–144.

Kay, M. (1980). The proper place of men and machines in language translation. *CSL*.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C., Bojar, O., Constantin, A., and Herbst, E. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lavie, M. D. C. D. A. (2014). Learning from post-editing: Online model adaptation for statistical machine translation. *EACL 2014*, page 395.

Och, F. J. and Ney, H. (2000). Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 440–447, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ortiz-Martínez, D., García-Varea, I., and Casacuberta, F. (2010). Online learning for interactive statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 546–554, Stroudsburg, PA, USA. Association for Computational Linguistics.

Pelletier, F. J. (1994). The principle of semantic compositionality. *Topoi*, 13(1):11–24.

Ranta, A. (2004). Grammatical framework. *Journal of Functional Programming*, 14(02):145–189.

Roturier, J., Mitchell, L., and Silva, D. (2013). The accept post-editing environment: A flexible and customisable online tool to perform and analyse machine translation post-editing. In *Proceedings of MT Summit XIV Workshop on Post-editing Technology and Practice*, pages 119–128.

Sato, S. and Nagao, M. (1990). Toward memory-based translation. In *Proceedings of the 13th conference on Computational linguistics-Volume 3*, pages 247–252. Association for Computational Linguistics.

Somers, H. (1999). Review article: Example-based machine translation. *Machine Translation*, 14(2):113–157.

Streiter, O. (1996). *Linguistic modeling for multilingual machine translation.*

Terumasa, E. (2007). Rule based machine translation combined with statistical post editor for japanese to english patent translation. In *Proceedings of the MT Summit XI Workshop on Patent Translation*, volume 11, pages 13–18.

Toselli, A. H., Vidal, E., and Casacuberta, F. (2011). Interactive machine translation. In *Multimodal Interactive Pattern Recognition and Applications*, pages 135–152. Springer.

Vogel, S., Zhang, Y., Huang, F., Tribble, A., Venugopal, A., Zhao, B., and Waibel, A. (2003). The cmu statistical machine translation system. In *IN PROCEEDINGS OF MT SUMMIT IX*, pages 110–117.

Wang, Y.-Y. and Waibel, A. (1998). Fast decoding for statistical machine translation. In *ICSLP*.