Master thesis planning report

Automating Concept Alignment for Machine Translation

Arianna Masciolini (`gusmasar@student.gu.se`)

Supervisor at CSE: Aarne Ranta

Relevant completed courses:

(LT2214, Computational Syntax)
(LT2003, Natural Language Processing)
(DIT143, Functional Programming)
(DIT260, Advanced Functional Programming)
(DIT231, Programming Language Technology)
(DIT300, Compiler Construction)
(DIT411, Introduction to Artificial Intelligence)
(DIT866, Applied Machine Learning)

October 22, 2020

1 Introduction

Concept Alignment (CA) consists in finding semantical correspondences between parts of parallel texts in two or more different languages, of which the Rosetta Stone is a notable example. Such task, which is routinely performed by learners of classical languages, who generally work with a translation alongside the original text, is often preliminary to further linguistic analysis.

Another task that involves CA is natural language translation: the human translator, almost unconsciously, first identifies concepts in the source text and then looks for ways to render them in the target language. In this case, alignment can happen - and usually does happen simultaneously - at different levels of abstraction, ranging from word to sentence level. It is then natural to wonder wether it is possible to make use of CA in Machine Translation (MT) as well. The hypotheses motivating this project, whose objective is to develop and test strategies for automating CA, is that MT can benefit from CA in two ways:

- on the one hand, if we aim to design a *compositional* MT pipeline, CA at some level is a necessary step, as will become clearer in the following sections

- on the other hand, if we address the problem from the perspective of XMT (eXplainable MT), providing the user of a MT application with a way to compare pairs of concepts instead of the entire text in the source language to its automatically translated counterpart can help building a more easily interpretable, and therefore more reliable MT system.

2 Problem

To give a more rigorous definition of CA, it is first necessary to specify what we mean by *concept*.

2.1 Concepts

Intuitively, concepts are the components of meaning, and therefore, in a multilingual context, the units of translation. The principle of semantic compositionality states that the meaning of a complex expression is a function solely of the meanings of its components and of the manner in which these components are combined [18]. If we assume this principle to be valid and apply it to translation, these meaning components, which we refer to as concepts, are the common denominator between an expression in the source language and its translation, which can be generated using them as the starting point.

We can draw a very close parallel between MT and compiler pipelines: in this sense, concepts are the intermediate representation, or *abstract syntax*, obtained by analyzing, or *parsing*, the source language, while we will call *backend* the portion of the pipeline aimed at *generating* the target language by applying the concepts a set of rules.

In this perspective, then, concepts are abstract syntax functions. In the simplest case, when they correspond to individual words, their arity is zero, but it is often the case that the concrete expressions corresponding to minimal translation units are, in one or both of the source and target language, lemgrams, multiword expressions - sometimes discontinuous - or even more complex constructions, whose intermediate representation is an abstract syntax function taking one or more arguments.

2.2 Alignment

Having defined concepts more accurately, we can turn to the more concrete problem of CA, the task of finding, at any abstraction level, correspondences between parallel and, in the case of translation, multilingual texts.

In order to facilitate its automation, CA can be divided into two subproblems subsequent to each other:

1. *concept creation*, the task of identifying concepts via language comparison, which is the main focus of this project

2. *concept propagation*, i.e. the task of finding, in a language which was not used in the concept creation stage, the concrete expressions corresponding to previously identified concepts.

3 Related works

The idea of taking advantage of semantic compositionality is not new in the field of MT. Introduced by Curry in the early 1960s [7], it was first put in practice two decades later in the form of an experimental interlingual translation system, not coincidentally named Rosetta, which requires the definition of two distinct logically isomorphic *Montague grammars* - one for the source language, one for the target language - and constructs an intermediate representation based on such isomorphism [11].

Montague grammars are a semantics-oriented development of *categorial grammars*, which in turn belong to *phrase structure* or *constituency* grammars, the class of grammatical formalisms based on the *constituency relation*. Without, at least for the time being, giving a formal definition of phrase structure grammars, we can say that they consist in a series of *rewrite rules*, such as

$$S \rightarrow NP\ VP,$$

which we can read as "*S* can be *rewritten* (or *substituted with*) *NP* and *VP*", or "a sentence *S* has a noun phrase *NP* and a verb phrase *VP* as its constituents".

Replacing the term "constituents" - as in *constituency relation* - with "components" makes it easy to understand in which way phrase structure grammars relate to the aforementioned principle of compositionality.

Among constituency grammars, other formalisms that have been employed in more recent MT systems are that of *synchronous CFG*, originally developed for programming language compilation [2] and adapted to natural language translation in several settings [21, 22, 6], and, later on, that of *synchronous TAG*, a variation of TAG (Tree-Adjoining Grammars) [9]. Both formalisms are meant precisely to characterize correspondences between languages by having as their elements, instead of single rewrite rules, pairs of rules - one for the source and one for the target language. In a synchronous TAG, the constituents of a source language rule may be linked to their counterparts in the corresponding target language rule, and such links, as the authors of [16] seem to suggest in section 4.1 on idiomatic expressions, may be used to identify concepts in a syntax-based fashion.

3.1 Grammatical Framework

Grammatical Framework (GF), a grammar formalism and programming language designed with the parallel between compilers and MT systems discussed in the previous section (2.1) in mind, goes a step further in this direction: it introduces a clear distinction between the abstract syntax, a phrase structure tree-like representation capturing the syntactic structures all natural languages taken into account have in common, and the *concrete syntaxes*, specific to each individual language, which consist in rules for the linearization of these Abstract Syntax Trees (ASTs) [12] [13]. This makes it possible to deal with multiple languages by writing only one grammar, whose components are an abstract syntax and several concrete syntaxes.

GF grammars can be used in a variety of contexts, one of which is interlingual MT. More in particular, experiments with GF grammars have been conducted in the field of XMT, as the ASTs produced by source language analysis can serve both as to some extent automatically checkable certificates for the correctness of translation - by backlinearization to the source language - and as fully inspectable explanations aimed towards expert users [14].

When it comes to CA itself, which could potentially provide more user-friendly explanations, the idea of performing syntax-based alignment by comparing GF ASTs seems promising, especially if compared with standard statistical approaches: *word alignment*, consisting in finding pairs of individual words that translate to each other, and its generalization, *phrase alignment*[1] [4]. In practice, however, comparing GF ASTs presents a number of issues related to the fact that GF parsing is not robust enough to spelling and grammatical errors,

[1]it must be noted that, in the context of statistical MT, the term "phrase" refers to nothing more than contiguous multiword segments

unusual word orders and rare constructions in general [14], thus making it hard, in many cases, to obtain trees that are actually useful for the task at hand.

It becomes then necessary, to perform syntax-based alignment, to make use of alternative parsers. Preliminary experiments, not yet published, which we intend to take forward in this project, have been conducted exploiting the similarity between GF and UD (Universal Dependencies) [10, 15].

3.2 Universal Dependencies

UD is a framework for cross-linguistically consistent grammatical annotation. The UD project aims at developing parallel treebanks for many languages in order to support, among other things, the development of multilingual *dependency parsers.*

Existing dependency parsers, such as UDPipe [17] and the Standford parser [5], are often - but not always [3] - neural pipelines trained on dependency treebanks that, given raw text as input, output *dependency trees.* The difference between such trees and those produced by GF parsing - and in general by natural language parsers based on phrase structure grammars - is that, while the latter capture *constituency* relations, dependency trees are based on another kind of relation, namely that of *dependency.* As opposed to constituency, dependency is a one-to-one correspondence, meaning that words are simply put in relation with each other via directed links. This relation serves as the basis for a comprehensive theory of grammar, first proposed in [19], alternative to that of phrase structure.

If phrase structure trees - and GF trees specifically - seem to be a good starting point for natural (target) language generation, replacing them with dependency trees - and, in particular, given their similarity with GF and the available parsers, UD trees - in the analysis step can prove to be a good basis for syntax-based CA, since the above mentioned UD parsers are more reliable than their GF-based counterparts. In more concrete terms, if, as mentioned in the introduction, CA is seen as a step of compositional MT, a UD parser can be integrated in a GF-based pipeline, thus constructing a hybrid translation system, with a robust neural frontend and a phrase structure grammar-based backend. In this case, dependency trees need to be converted into GF ASTs, albeit with the disadvantage that, unlike its reverse [10], the UD-to-GF conversion is a non-deterministic search problem, addressed in [15], where `gfud`, a program for converting UD trees into GF ASTs - and vice versa - is presented.

4 Objectives

The overall goal of this project is to test and further develop the above mentioned ideas for CA automation.

Nonetheless, a first, necessary step consists in selecting a UD parser well suited to the task. Once an adequate parser is identified, the possibility of tuning it for the task at hand will be considered: while our direct intervention on its source code is unlikely, as it goes beyond the scope of this project, it could prove useful to configure it and/or, in the case of a machine learning-based parser, to train it on a suitable corpus.

When it comes to syntax-based alignment itself, which is our proposed way to address concept creation and as such the core part of this project, an unpublished algorithm for aligning pairs of dependency trees exists, but there is, in many respects, room for improvement: preliminary experiments show that its precision is still very low, and there's a number of specific known issues to work on, as the lack of a way for it to recover from parse errors and to deal with irrelevant cross-lingual differences. A central part of this project will then be improving the existing program in terms of both precision and robustness.

Once significant improvements have been achieved, it is our intention to incorporate the results of the alignment program - the concepts in the form of a GF grammar - into a domain-specific MT pipeline, whose stages are:

1. the dependency parser whose selection is the first step of this project
2. `gfud` for converting the dependency trees obtained at step 1 into GF ASTs
3. a GF-based frontend for target language generation using the domain-specific grammar defined starting from concepts

Such system, also illustrated in figure 1, will be, together with the report, the final output of this project and provide a way to evaluate the system, as explained more thoroughly in section 5.2.6.

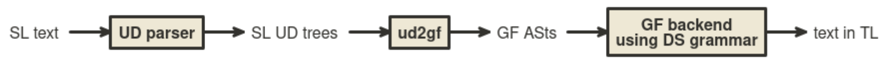

Figure 1: The domain-specific MT system we aim to ahve as the final output of this project. SL stands for "Source Language", TL for "Target Language".

4.1 Stretch goals

If time allows it, a natural extension to this project is to consider more than just two languages, as results may differ significantly for different source-target language pairs. Taking a third language into account can be a way to:

- enhance concept creation, for instance by generalizing the tree alignment program to work on arbitrary-length lists - instead of pairs - of trees

- address concept propagation, by trying to find the equivalent of concepts obtained by aligning trees in the original language pair in the newly introduced language.

Our criteria for choosing which languages to work with are described in the following section.

5 Approach

5.1 Data

Building upon previous experiments, we will conduct a case study on the text of the General Data Protection Regulation (GDPR) [1]. As all European regulations, it is a very multilingual parallel text, as it has a high quality translation for each of the 24 official languages of the European Union. Furthermore, for 5 of these languages (English, German, Spanish, Italian and French), a gold standard in terms of concept extraction, even though not publicly available, already exists in the form of a domain-specific GF lexicon.

There is a risk, however, that the complexity of such text, which is characterized by very long sentences, is too high even for a state-of-the-art dependency parser. As a consequence, we reserve ourselves the possibility to work, at least in the initials stages of the project, on alternative corpora. One that would be easily available thanks to a recent thesis project is, despite some limitations such as being no more than bilingual, that of the course plans of some departments of the University of Gothenburg.

Furthermore, another corpus we plan to employ for preliminary testing of our tree alignment program is that of Parallel UD (PUD) treebanks, developed within the UD Project.

5.1.1 Languages

Our choices in terms of which languages to address in this project are of course also limited by the linguistic knowledge of the candidate. As for the initial language pair to work with, we intend to use Italian and English, which, apart from being respectively her first and second language, belong to two different branches - resp. Romance and Germanic - of the Indo-European language family, making CA a potentially harder, but at the same time more interesting task. In the aforementioned case that we decide to also perform experiments with one or more other languages, it would be possible to work with yet another one for which we already have a gold standard available, Spanish, and/or we could consider using Swedish, which would also require the manual creation of a new GDPR lexicon.

5.2 Work plan

In this section, we present our plan for the project, in the form of a description of what is to be done at each stage followed by our proposed time plan (cf. section 5.3).

5.2.1 Dependency parsing

As mentioned above, the initial phase of the project will be dedicated to identifying a suitable UD parser. The choice will be based on an informal evaluation of the dependency trees obtained for one or more bilingual parallel texts and will be followed by configuration and, if applicable and necessary, retraining of the chosen model. The expected output of this step is a parallel treebank relative to the corpus (or corpora) in question.

5.2.2 Dependency tree alignment program

The syntactic alignment component will be based on an initial existing Haskell prototype. While we do plan to work on the issues we are already aware of, listed in section 5.2.4, what other specific problems we address and how exactly we decide to proceed to obtain a substantially improved version of the existing program will mostly depend on the results obtained by testing the prototype on the aforementioned PUD treebank. Being this the fundamental part of the project, we plan to work on it for at least one third of the time available.

5.2.3 Preliminary evaluation

Together, the dependency parser and the tree alignment program allow concept creation, the fundamental prerequisite for the backend of our desired MT system. Such component, once ready, will be tested independently from the rest of the MT pipeline. This preliminary evaluation will consist for the most part in assessing the precision of the concepts extracted via the tree alignment module against a human-made gold standard, which is, as mentioned, already available for the GDPR corpus as a GF lexicon.

Furthermore, a preliminary MT evaluation, in terms of individual concepts, will already be possible at this stage. In particular, for single-word concepts, a way to make sure that a lemma in the source language corresponds its target language counterpart is sense comparison using Universal WordNet [8] data, which have already been successfully used in GF-based MT systems [20].

5.2.4 From UD to GF

Once the preliminary evaluation is concluded, the results of concept creation can be incorporated in a full MT pipeline in the form of a GF abstract lexicon and two GF concrete lexica, one for English and one for Italian. Automating this step means converting matching UD trees representing concepts to GF ASTs.

Work in this sense will involve intervening on or at least configuring another existing program, `gfud`, already mentioned in section 3.2. Finally, these GF trees need to be converted into an actual lexicon, ideally via a yet-to-be written GF code generation module. Once this step is completed, since the target language concrete syntax of the lexicon specifies the linearization rules necessary for generating the translation of a sentence in the source language, all the stages of the MT pipeline illustrated in figure 1 and described in section will be ready and easy to put together in a complete domain-specific MT system.

5.2.5 Introduction of a third language

Once a full working pipeline for concept creation exists, given the multilingual nature of both UD and GF, it is in principle not too demanding to add other languages. As mentioned above, what will be done with respect to this and even the choice of the language itself will depend on how much time we will still have available. However, some ideas are mentioned in section 4.1.

5.2.6 Overall evaluation

While the strategies for the overall evaluation of the system resemble, for the most part, those described in section 5.2.3, given that at this point we will have a fully functional MT pipeline, it will be possible to translate entire passages from one language to another and vice versa to compare the original text in the source language to a version of it obtained by linearizing its translation to the target language back in the source language again, or to compare the automatically generated translations with the official ones.

After a first test on one or more of the aforementioned corpora, it will be necessary, in order to assess the generalization capabilities of our system to test it on other parallel texts. Assuming we stick to using the text of the GDPR, as previously pointed out, the Official Journal of the European Union is itself a relatively large, highly multilingual parallel corpus of legal documents, but, even if good results within this context would already be meaningful, it would be interesting to put the system to the test - after performing concept alignment again - in other domains, for instance using public domain literature.

5.3 Time plan

The following table describes the timeline of the project. More or less specific objectives both in terms of implementative work and writing have been set on a weekly basis, in accordance with the externally imposed deadlines. The general idea is to write the different sections of the final report as soon as the corresponding experiments are performed, while dedicating the last week to the adjustments and integrations which will prove necessary.

It should be noted that:

- the two weeks during which this document has been written are also part of the time plan. At the time of writing, week 40, as the table itself suggests, the dependency parser to be used has already been selected and is being tested on the GDPR Italian-English parallel text in order to find a suitable configuration and to identify issues to be solved either by additional data preprocessing or to be kept in mind when implementing later stages of the pipeline

- in this time plan, we assume that we will be able to reach, at least to some extent, the goal of working in a trilingual settings. As a consequence, especially in its final part, the plan might be subject to changes

- this time plan was written under the assumption that the only presentation date available in January is the 19th. However, we think it would be preferable, if possible, to postpone the presentation by 7-10 days so to have a one week break in week 52 or 53.

Week	**Deadlines**	**Experimental work**	**Writing**
39/'20		dependency parsing (selection and installation)	work and time plan
40/'20		dependency parsing (tuning and application)	related works, dependency parsing, completion of planning report
41/'20	planning report	tree alignment program	
42/'20		tree alignment program	
43/'20		tree alignment program	initial tree alignment program overview
44/'20		tree alignment program	tree alignment program: proposed improvements
45/'20		tree alignment program	tree alignment program: proposed improvements
46/'20		preliminary evaluation	evaluation strategies, preliminary results
47/'20		gfud	relation between GF and UD, `gfud`
48/'20	halftime report	gfud	work done in relation to `gfud`
49/'20		gfud	work done in relation to `gfud`
50/'20		introduction of a third language	introduction of a 3rd language
51/'20		overall evaluation	more on evaluation strategies
52/'20		overall evaluation	results
53/'20			updated introduction, conclusions, finishing touches
1/'21	final report		
2/'21	presentation		

References

[1] General data protection regulation. *Official Journal of the European Union.*

[2] JW BACKerS et al. Ano, av [1968]. indexed grammars˜ an extension of context-free grammars. jr., 4cm 15: 4, 647-671. aho, av, and jd ullman [1969a]. syntax directed translations and the pushdown assembler. j. computer and system sciences 3: 1, 37-56. ano, av, and jd ullman [1969b]. properties of syntax directed translations.

[3] Ted Briscoe, John A Carroll, and Rebecca Watson. The second release of the rasp system. In *Proceedings of the COLING/ACL 2006 interactive presentation sessions*, pages 77–80, 2006.

[4] Peter F Brown, Stephen A Della Pietra, Vincent J Della Pietra, and Robert L Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311, 1993.

[5] Danqi Chen and Christopher D Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750, 2014.

[6] David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (acl'05)*, pages 263–270, 2005.

[7] Haskell B Curry. Some logical aspects of grammatical structure. *Structure of language and its mathematical aspects*, 12:56–68, 1961.

[8] Gerard De Melo and Gerhard Weikum. Towards a universal wordnet by learning from combined evidence. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 513–522, 2009.

[9] Aravind K Joshi and Yves Schabes. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer, 1997.

[10] Prasanth Kolachina and Aarnte Ranta. From abstract syntax to universal dependencies. In *Linguistic Issues in Language Technology, Volume 13, 2016*, 2016.

[11] Jan Landsbergen. Machine translation based on logically isomorphic montague grammars. In *Coling 1982: Proceedings of the Ninth International Conference on Computational Linguistics*, 1982.

[12] Aarne Ranta. Grammatical framework. *Journal of Functional Programming*, 14(2):145–189, 2004.

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

[14] Aarne Ranta. Explainable machine translation with interlingual trees as certificates. In *Proceedings of the Conference on Logic and Machine Learning in Natural Language (LaML 2017)*, pages 63–78, 2017.

[15] Aarne Ranta and Prasanth Kolachina. From universal dependencies to abstract syntax. In *Proceedings of the NoDaLiDa 2017 Workshop on Universal Dependencies (UDW 2017)*, pages 107–116, 2017.

[16] Stuart Shieber and Yves Schabes. Synchronous tree-adjoining grammars. In *Proceedings of the 13th international conference on computational linguistics*. Association for Computational Linguistics, 1990.

[17] Milan Straka, Jan Hajic, and Jana Straková. Udpipe: trainable pipeline for processing conll-u files performing tokenization, morphological analysis, pos tagging and parsing. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4290–4297, 2016.

[18] Zoltán Gendler Szabó. Compositionality. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, fall 2020 edition, 2020.

[19] Lucien Tesnière. Éléments de syntaxe structurale. 1959.

[20] Shafqat Mumtaz Virk, KVS Prasad, Aarne Ranta, and Krasimir Angelov. Developing an interlingual translation lexicon using wordnets and grammatical framework. In *Proceedings of the Fifth Workshop on South and Southeast Asian Natural Language Processing*, pages 55–64, 2014.

[21] Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational linguistics*, 23(3):377–403, 1997.

[22] Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 523–530, 2001.